il BUGBOOK® VII

L'INTERFACCIAMENTO

FRA **MICROCOMPUTER-**

E **CONVERTITORI ANALOGICI**

HARDWARE, SOFTWARE,
ESPERIMENTI PER SISTEMI 8080/A - Z80 - 8085

EDIZIONE
ITALIANA

J.A TITUS
P.R. RONY
C.A. TITUS
D.G. LARSEN

JACKSON
ITALIANA
EDITRICE

Bugbook ® è un marchio registrato della E&L Instruments, Inc. di Derby, Connecticut. Esso è relativo ad una serie di libri concepiti e realizzati per un approccio di tipo sperimentale ai problemi di elettronica. I libri di questa serie, in continua espansione, comprendono:

Bugbook I° & II° - Principi di elettronica digitale con esercizi applicativi ed uso pratico dei circuiti integrati della serie 7400.

Bugbook II°A - Tecniche di comunicazione asincrone digitali. UART e loops di corrente 20 mA, con esperimenti.

Bugbook III° - Interfacciamento e programmazione del microprocessore 8080.

Bugbook IV° - Chip d'interfaccia 8255 (in preparazione).

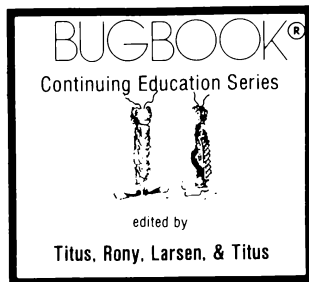
Bugbook V° & VI° - Introduzione ai micro con ripasso dei concetti base di elettronica digitale. Programmazione e interfacciamento dell'8080A, con esperimenti ed esercizi.

Bugbook VII° - Interfacciamento dei microcomputer coi convertitori analogici: software e hardware, con esperimenti.

Bugbook VIII° - Sulla programmazione dei micro 8080 e 8085 e i seguenti testi della serie **"Bugbook Reference Series":**

- Il timer 555, funzionamento, applicazioni ed esperimenti
- La progettazione dei filtri attivi, con esperimenti
- DBUG: un debugger interpretativo per l'8080
- La progettazione degli amplificatori operazionali, con esperimenti
- Corso di elettronica di base della NCR, con esperimenti
- Elementi di trasmissione dati (a cura della NCR)
- Phase locked loops, con esperimenti

Ciò che maggiormente caratterizza i Bugbook ® sono la chiarezza di esposizione e la descrizione dettagliata degli esperimenti, appositamente studiati per fornire allo studioso la possibilità di una verifica sperimentale dei temi trattati. I Bugbook ® integrano infatti teoria e sperimentazione e sono tra i pochissimi testi che garantiscono un tipo di istruzione programmata sui concetti fondamentali dell'elettronica. Essi descrivono l'utilizzazione pratica di circuiti integrati sia lineari che digitali, spiegano come interfacciare un microcomputer basato sull'8080, come memorizzare, provare ed eseguire programmi su questo microcomputer. L'esperienza ci conferma che questi libri sono di grande aiuto per coloro che desiderano partecipare alla "rivoluzione dell'elettronica" attualmente in atto, disponendo di tempo limitato e impiegando minimo sforzo e minima spesa.
I Bugbooks hanno riscosso un notevole successo negli Stati Uniti e all'estero. Alcuni di essi sono stati tradotti in Italiano, Tedesco, Giapponese e Malese. Allo stesso modo i nostri corsi sui microcomputer e l'elettronica digitale, che pure hanno riscosso notevole successo negli Stati Uniti, vengono ora richiesti e organizzati in altri paesi. Questa nostra attività si inquadra in quella del Continuing Education Center and Extension Division del Virginia Polytechnic Institute & State University, per cui potrete richiedere ulteriori informazioni al Dr. Norris H. Bell, presso l'Università o telefonando negli USA al numero (703) 951-9030.
In Italia l'attività della Continuing Education é stata iniziata e condotta con successo ad opera della MICROLEM divisione didattica, Via C. Monteverdi 5, 20131 MILANO Tel. (02) 27 10 465, che, oltre a collaborare strettamente con i maggiori editori europei per la pubblicazione dei Bugbooks, organizza corsi brevi sui microcomputer con la nostra collaborazione ed il nostro diretto intervento.

David G. Larsen, Peter R. Rony, Jonathan A. Titus, and Christopher A. Titus
"The Blacksburg Group"
Blacksburg, Virginia 24060

il BUGBOOK® VII

L'INTERFACCIAMENTO

FRA **MICROCOMPUTER-**

E **CONVERTITORI ANALOGICI**

HARDWARE, SOFTWARE,
ESPERIMENTI PER SISTEMI 8080/A - Z80 - 8085

di
Jonathan A. Titus
Christopher A. Titus
David G. Larsen

JACKSON ITALIANA EDITRICE
Piazzale Massari, 22 - 20125 Milano

PREFAZIONE ALL'EDIZIONE ITALIANA

Testi che trattino *applicazioni* dei microprocessori non sono attualmente molto diffusi in Italia. Questa lacuna, particolarmente sentita sia dai tecnici esperti che dai neo-progettisti, è validamente colmata da questo libro, che bene si inserisce nella collana dei Bugbooks pubblicati in Italia dalla Jackson Editrice.

Il Bugbook VII riveste particolare interesse fra i testi della collana in quanto, utilizzando concetti ed esperimenti descritti nei precedenti testi, in particolare nei Bugbook V e VI, permette di capire come un sistema a microprocessore si interfacci con il mondo esterno. A questo scopo, una particolare attenzione è rivolta ai convertitori, sia digitali-analogici che analogici-digitali, dei quali viene ampiamente discusso l'aspetto software oltre che i problemi di interfacciamento, grazie alla descrizione di circuiti applicativi e di programmi.

Anche in questo testo, come in altri della collana, la descrizione degli esperimenti, è parte integrante della materia e costituisce verifica degli argomenti trattati. La loro esecuzione è possibile, con modica spesa, utilizzando un microcomputer didattico MMD 1, un "breadboard" SK 10, alcuni "outboard", tra i quali il "digital to analog converter" LR 35, ed alcuni componenti, come descritto nella parte introduttiva del capitolo 7. Tutti questi sussidi si possono reperire facilmente presso i migliori rivenditori specializzati.

Anche questo volume, nell'ottima traduzione della Mipro e nella impeccabile veste tipografica della Jackson, è destinato a riscuotere il successo di quelli che l'hanno preceduto. Auguriamo che esso possa soddisfare appieno le aspettative di quei tecnici e autodidatti che vogliono approfondire le proprie conoscenze nel campo delle applicazioni dei sistemi a microprocessore.

MICROLEM divisione didattica — *Milano, 1980*

INTRODUZIONE

Il nostro scopo, nello scrivere questo libro, è stato quello di introdurvi nei concetti e nelle tecniche di interfacciamento dei microcomputer con dispositivi elettronici dualogici. I concetti presentati in questo libro possono essere probabilmente estesi a metodi di interfacciamento e di programmazione validi per molti tipi di microcalcolatori. Abbiamo concentrato la trattazione sull'interfacciamento di convertitori sia analogico-digitali che digitali-analogici con microcalcolatori basati sui microprocessori 8080A e 8085; i progetti e i programmi sono comunque utilizzabili indistintamente su tutti i microcalcolatori utilizzanti i microprocessori 8080, 8080A, 8085, Z-80.
Non è nostro scopo entrare nei dettagli del funzionamento interno dei moduli di conversione analogici e digitali, ma piuttosto li considereremo come dispositivi elettronici come gli elementi della serie TTL SN7400 o CMOS CD4000.

Non affronteremo l'argomento della costruzione di convertitori analogico-digitali o digitali-analogici, avendo scelto invece di mostrarvi l'uso di moduli e dispositivi che sono normalmente disponibili sul mercato. Vi presenteremo molti esempi di interfacciamento, completi di schemi elettrici e listing dei programmi e probabilmente uno di questi progetti vi potrà essere utile in qualche applicazione, facendovi risparmiare lavoro e fatica. Se invece le vostre esigenze riguardano situazioni specifiche, speriamo che possiate trovare dei concetti che vi agevolino nella vostra attività di progettazione. Se sarà così avremo raggiunto il nostro scopo.

Abbiamo fatto l'ipotesi che da parte vostra sia possibile condizionare (amplificare, filtrare, ecc.) in modo opportuno i segnali analogici, cosicché siano compatibili con i convertitori A/D utilizzati negli esempi, e che possiate utilizzare le tensioni in uscita dei dispositivi stessi.
Gli argomenti quali il *condizionamento dei segnali,* il *filtraggio,* i *disturbi,* i *loop di terra* e gli *amplificatori,* sono al di fuori dallo scopo di questo libro; saranno oggetto di uno dei prossimi Bugbook.

Se è vostro interesse approfondire il funzionamento interno dei convertitori A/D e D/A, vi sono due ottimi volumi che affrontano l'argomento:

> *Analog-Digital Converters Notes,* D.H Sheingold, ed., Analog Device, Inc. Norwood, MA 02062, 1977;
>
> *Data Conversion Handbook,* Donald B. Bruck, Hybrid Sistems Corporation, Burlington, MA 01803, 1974.

In tutti gli esempi abbiamo cercato di utilizzare moduli e circuiti integrati tra i più diffusi anche se ci sono parecchi costruttori di moduli e dispositivi di conversione analogico/digitali; alcuni di questi sono illustrati nei capitoli seguenti o in appendice. Le nostre scelte sono state effettuate casualmente, cercando di individuare i moduli più semplici da utilizzare e che fornissero schemi di applicazione.

La scelta non è stata quindi dettata da particolari interessi e non deve neanche essere interpretata come la segnalazione dei dispositivi migliori disponibili sul mercato, con un conseguente ovvio messaggio promozionale.

Si suppone che il lettore abbia una certa esperienza di programmazione in linguaggio assembler su microcalcolatori e che conosca l'8080, l'uso dei vari registri e le operazioni relative. Saranno affrontate sia tecniche di I/O sia con mappaggio sulla memoria che tramite l'accumulatore, con una certa preferenza per quest'ultimo metodo per la sua semplicità. E' necessaria una certa familiarità con l'uso di latch come parte di uscita, di buffer three-state come porta in ingresso e l'uso dei decodificatori per la decodifica degli indirizzi dei vari dispositivi. Tutti gli esempi e le applicazioni suppongono l'utilizzo di un sistema 8080 con un bus di dati bidirezionale non invertente.

I programmi riportati negli esempi e nel libro presentano il formato di "un byte per linea" così come viene prodotto dal Tychon Editor/Assembler (TEA) scritto dal Dr. Christopher A. Titus, in quanto riteniamo sia più facile da leggere e interpretare che non il formato di "un'istruzione per linea", di altri assembler. Inoltre, sebbene i nostri favori vadano ad una rappresentazione con sistema di numerazione ottale, il programma TEA può anche fornire formati esadecimali.

I bugbook hanno incontrato finora un largo consenso sia da parte di aziende che di singoli utenti; infatti è già stata effettuata la traduzione in molte lingue come Tedesco, Giapponese, Francese, Cinese, Malese e naturalmente Italiano.Nel caso in cui vi interessassero maggiori dettagli relativi a queste traduzioni, Vi preghiamo di mettervi in contatto con la casa editrice Jackson Italiana.

La serie dei Bugbook continua ad espandersi con le edizioni degli ultimi mesi e oggi la serie dei Bugbook è composta dai seguenti volumi:

Bugbook I e II, Esperimenti su circuiti logici e di memoria, D.G. Larsen P.R Rony.

Bugbook IIA, Esperimenti di interfacciamento e comunicazione dati, D.G Larsen e P.R. Rony.

Bugbook III, Interfacciamento di sistemi a microcomputer, P.R. Rony, D.G Larsen e J.A. Titus.

Bugbook IV e V, Esperimenti introduttivi all'elettronica digitale del microprocessore 8080A, Rony, Larsen, Titus.

Tutti questi volumi sono stati tradotti in italiano dalla MIPRO S.r.l. e sono editi dalla JACKSON ITALIANA EDITRICE.

Inoltre esiste una seconda serie di Bugbook orientata a specifici argomenti di elettronica. Questa serie porta il nome di Bugbook Application Series ed è attualmente formata dai seguenti volumi:

Il timer 555: Applicazione ed Esperimenti;
Howard M. Berlin

La progettazione dei Filtri Attivi, con Esperimenti;
Howard M. Berlin;

DBUG: An 8080 Interpretive Debugger, C.A. Titus e J.A. Titus;

La progettazione dei circuiti Amplificatori Operazionali, con Esperimenti.
Howard M. Berlin.

Attualmente il Dr. Stanley Wolf sta scrivendo un nuovo volume, di quest'ultima serie, sulla teoria e l'uso dell'oscilloscopio mentre il Dr. Chris Titus è alle prese con un libro di programmazione di microcalcolatori particolarmente corredato di esempi di programmi. Siamo comunque sempre interessati a chiunque ritenga di avere delle idee per la realizzazione di volumi che affrontino problemi e argomenti che possono inserirsi nella serie dei Bugbook.

Molti dei concetti esposti in questo libro sono parte delle trattazioni che vengono effettuate nei corsi sui microcalcolatori e microprocessori, realizzati in Italia dal Virginia Polytechnic Institute e State University e dalla Mipro S.r.l. di Milano con l'organizzazione dello Studio C.P.M. e la sponsorizzazione della Microlem.

Per tutti coloro che siano interessati a questi corsi, diamo di seguito una serie di indirizzi cui fare riferimento:
Studio C.P.M., Via Melchiorre Gioia 55 Milano - Tel. 683680
Mipro S.r.l., Sezione Didattica, Via Carducci, 15 Milano Tel. 897151, 879067. Sig.ra Cavenaghi
Microlem, Via C. Monteverdi, 5 Milano - Tel. 220326 - 220317. Sig. Gandini.

La realizzazione di questo volume è stata possibile anche grazie a numerose case che ci hanno fornito informazioni, assistenza tecnica e possibilità di riproduzione di loro specifiche tecniche e Libri. Ringraziamo quindi la ADAL Corporation, Analog Devices, Burr-Brown, Data Translation, Datel, National, Signetics, Siliconix e Texas Instruments.I nostri ringraziamenti vanno inoltre a Mr. Munay Gallant della E & L Instruments per l'assistenza costante nella stesura del libro.

SOMMARIO

CAPITOLO 4 - MISCELLANEA DI TECNICHE DI CONVERSIONE

CAPITOLO 5 - CIRCUITI DI SAMPLE-AND-HOLD E DISPOSITIVI MULTIPLEXER

CAPITOLO 6 - ... DEI BIT, SCHEDE E SCATOLE NERE

CAPITOLO 7 - ESPERIMENTI CON CONVERTITORI DIGITALI-ANALOGICI E ANALOGICO-DIGITALI

APPENDICE N. 1 SPECIFICHE DEI CONVERTITORI A/D e D/A

APPENDICE N. 2 DATA SHEETS DI UNA SELEZIONE DI PRODOTTI, CONVERTITORI

CAPITOLO 1

INTERFACCIAMENTO DEI CONVERTITORI DA DIGITALE AD ANALOGICO

INTRODUZIONE

La maggior parte dei convertitori digitale-analogico (DAC o convertitori D/A) possono essere considerati come dispositivi che generano in uscita un segnale in corrente o in tensione; il valore di questo segnale è direttamente proporzionale al valore dell'ingresso digitale, generalmente in formato binario.In questo capitolo verrà presentato come interfacciare i convertitori D/A con i microcalcolatori e saranno mostrati alcuni esempi di uso di questi dispositivi. I convertitori D/A sono trattati come moduli di cui viene specificato il funzionamento interno; la comprensione della reale logica di funzionamento dei convertitori può essere interessante, ma non riteniamo che ciò possa aumentare la vostra esperienza per quanto riguarda problemi software e di interfacciamento.

OBIETTIVI

Dopo aver letto questo capitolo sarete in grado di:

- Calcolare il minimo incremento di tensione per un convertitore D/A a 11 bit con una variazione in uscita di x volt.
- Descrivere come realizzare l'interfacciamento di un semplice convertitore D/A a 8 bit.
- Descrivere come i convertitori D/A, possano essere utilizzati per generare i seguenti segnali in uscita:

  Rampe positive e negative (a dente di sega)
  Forme d'onda triangolari
  Onde quadre
  Forme d'onda complesse.

- Progettare una interfaccia per un convertitore D/A a 10 bit con doppia bufferizzazione.
- Scrivere un programma per il controllo di due convertitori D/A per la visualizzazione X-Y di un file di dati.

La maggior parte dei convertitori D/A fornisce, come abbiamo detto, un segnale in uscita che è proporzionale ad alcuni tipi di segnali digitali in ingresso. Tipicamente, questi segnali, hanno un codice binario e ciascun ingresso può avere solo uno dei due valori logici 1 o 0. Se consideriamo un convertitore D/A a quattro bit con gli ingressi A, B, C, D che hanno rispettivamente un peso 1, 2, 4, 8 avremo 16 combinazioni possibili dai valori 0000_2 a 1111_2 compresi.

Assumiamo che ciascun ingresso binario comporti un contributo in tensione al valore totale, proporzionale al suo peso binario. Le tensioni più semplici da usare in questo esempio sono A = 1 volt, B = 2 volt, C = 4 volt, D = 8 volt; se uno 0 è presente in una particolare colonna, il valore di tensione corrispondente non verrà aggiunto al totale, mentre se è presente un 1, sì.

In questo convertitore D/A a quattro bit la variazione di tensione sarà da 0 a 15 volt, così come è mostrato nella seguente tabella:

Codice binario in ingresso				Tensione in uscita
D=8	C=4	B=2	A=1	Volt
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9
1	0	1	0	10
1	0	1	1	11
1	1	0	0	12
1	1	0	1	13
1	1	1	0	14
1	1	1	1	15

La maggior parte dei convertitori D/A reali utilizzano una serie di resistori di valore R, 2R, 4R, 8R, in una combinazione binaria o reti resistive a maglie, di valore R-2R. La soluzione con la rete a maglie è preferibile in quanto è più facile la produzione e migliori sono le caratteristiche elettriche del dispositivo.
Le tensioni sono attuate elettronicamente, ma come ciò avvenga è al di fuori del nostro scopo.
Attualmente sono anche disponibili convertitori D/A che forniscono un'uscita in corrente: presentano il vantaggio di un *tempo* di *settling* particolarmente veloce, generalmente poche centinaia di nanosecondi.
Per un approfondimento dello studio dei convertitori D/A e di fattori come *linearità*, *offset*, e *risoluzione*, vi rimandiamo ai dati segnalati nell'introduzione, mentre questi termini tecnici ed altri sono definiti nell'Appendice 1.

I convertitori normalmente presentano degli ingressi compatibili TTL che facilitano l'interfacciamento sia con i circuiti della serie 7400 che con i circuiti integrati dei sistemi a microprocessore. Qualcuno dei più recenti presenta anche una compatibilità con i circuiti CMOS o ECL.

Sebbene nel corso delle nostre applicazioni adotteremo convertitori D/A che richiedono codici esclusivamente binari, è doveroso segnalare che esistono in commercio dei tipi che richiedono altri tipi di codici. Generalmente i codici disponibili sono:

- Complemento a due
- Binario complementato
- Offset binario complementato
- Offset binario
- Codice binario decimale (BCD)

Questi codici possono essere utili in molte applicazioni, ma abbiamo notato che i convertitori con codici binari sono quelli più utilizzati nell'interfacciamento con i microcalcolatori.

Tensioni di uscita di 0–10 volt, 0–5 volt, ±2,5 volt, ±5 volt e ± 10 volt sono i valori più comunemente usati e comunque la gamma dei convertitori D/A è così vasta da soddisfarvi, qualsiasi siano le vostre esigenze specifiche.

Nel caso di valori non standard è invece necessario l'uso di un amplificatore operazionale per "personalizzare" il campo di variazione del segnale di uscita. Difficilmente un utilizzatore è costretto al progetto di uno specifico convertitore D/A in considerazione del grande numero di circuiti disponibili.
Convertitori con ingressi di tipi digitale da 8, 10, 12 e 14 bit sono diffusi, con una quotazione che parte da circa 15 $ fino alle centinaia di dollari.

SEGNALI DI USCITA DEI CONVERTITORI D/A

Quando un ingresso digitale viene presentato in modo parallello a un convertitore D/A, quest'ultimo risponderà fornendo in uscita un valore di tensione proporzionale al valore digitale del segnale d'ingresso. Come è possibile determinare quale sarà il valore di questo segnale in uscita per un convertitore D/A a *n*/bit in un campo di variazione di X volt?

Si supponga di avere a disposizione un convertitore D/A a otto bit con codice binario in ingresso e variazione in uscita da 0 a 1 volt. Gli *n*/bit, in questo caso 8, forniscono 256 valori, da 0 a 255 compreso. L'uscita può essere divisa in 256 porzioni individuali, ciascuna di 0,00390 volt o più semplicemente 3,9 millivolt.
Possiamo quindi determinare il valore di tensione in uscita di un convertitore D/A per un certo valore di ingresso, X:

$$V_{out} = \text{Tensione in uscita} = (\text{1 volt di fondo scala}) \cdot \left(\frac{X}{256} \right)$$

oppure può essere usata la seguente formula:

$$V_{out} = \text{Tensione in uscita} = (\text{X porzioni}) \cdot (\text{0,00390 volt}).$$

Una tabella che mostra alcuni dei valori compresi tra 0 e 1 volt di fondo scala è mostrata di seguito:

Ingresso binario	Valore di tensione (in volt)
00000000	0,00390625
00000001	0,00781250
00000010	0,01171875
00000011	0,01562500
00000100	0,01953125
....	....
....	....
11111011	0,98046875
11111100	0,98437500
11111101	0,98828125
11111110	0,99218750
11111111	0,99609375

È importante notare che mentre il convertitore D/A usato in questo esempio ha un campo di variazione in uscita indicato da 0 a 1 volt, l'uscita analogica non raggiunge il valore preciso di 1 volt. La ragione di ciò vi sarà chiara se analizzate il "peso" in tensione assegnato a ciascuno degli otto bit: infatti in quanto si utilizza una numerazione binaria, ciascun bit è una potenza intera di due.
Così quando ci spostiamo dal bit più significativo posto all'estrema sinistra a quello meno significativo posto all'estrema destra, ciascuna posizione ha un "peso" che è metà del bit più vicino a sinistra e doppio di quello più vicino a destra.

Per concludere riportiamo l'elenco dei pesi dei singoli bit nel caso del fondo scala di 1 volt.

Bit meno significativo	0,00390625 = 1/256	volt
	0,00781250 = 1/128	volt
	0,01562500 = 1/64	volt
	0,03125000 = 1/32	volt
	0,06250000 = 1/16	volt
	0,12500000 = 1/8	volt
	0,25000000 = 1/4	volt
Bit più significativo	0,50000000 = 1/2	volt

GENERAZIONE DI SEGNALI ANALOGICI

I moduli di conversione D/A convertono velocemente il valore binario all'ingresso in tensione, ma il segnale all'ingresso deve essere mantenuto un tempo sufficiente per la generazione dell'uscita.

Se proviamo ad interfacciare un convertitore D/A ad un microcalcolatore semplicemente collegando gli otto ingressi con i bus dei dati, notiamo che il convertitore genera una tensione d'uscita che cambia continuamente. I diversi valori sono causati dal continuo variare della configurazione ai morsetti di ingresso durante il funzionamento del microprocessore che usa il bus per il trasferimento di dati e istruzioni. Non è dunque solo necessario presentare all'ingresso di un convertitore D/A il valore da convertitore, ma anche mantenerlo costante per il tempo necessario.

A questo scopo può essere utilizzato un semplice circuito latch per catturare l'informazione normalmente fornita tramite un bus dei dati di un microcalcolatore. I convertitori digitale-analogico generalmente richiedono poco più di una circuiteria di latch e decodifica oltre il convertitore, per completare l'interfacciamento.

Un tipico interfacciamento di un convertitore D/A è mostrato in Figura 1-1. In questo esempio è utilizzato il modulo di conversione a 8 bit DAC-98BI della DATEL SISTEMS, che fornisce un'uscita in corrente che deve essere convertita in tensione per ottenere il range da 0 a 1 Volt.

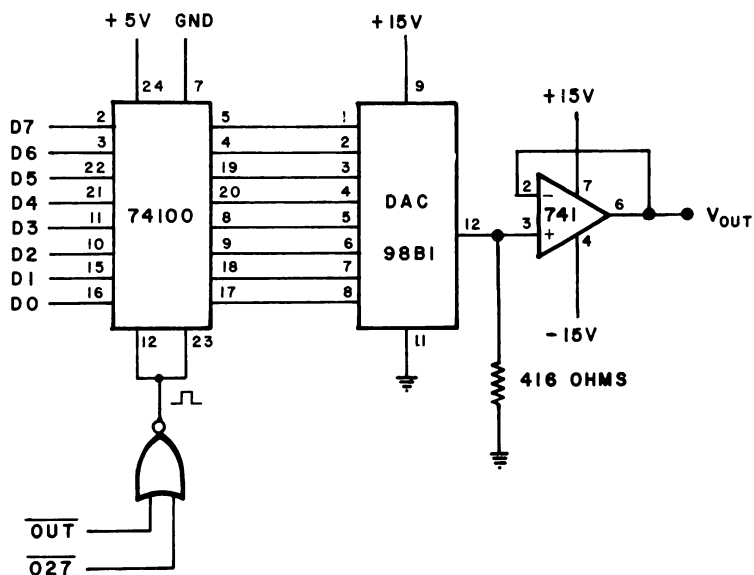



Figura 1-1. Interfacciamento del convertitore DATEL DAC 98BI con un sistema a microcalcolatore con l'uso di un circuito latch.

Per maggiori informazioni circa l'indirizzamento e la decodifica del dispositivo così come è stato riportato nella Figura 1-1 per l'attivazione del latch, vi rimandiamo al *Bugbook VI* Capitolo 17. (Il Bugbook VI è edito dalla Jackson Italiana).

Vi sono molti impieghi per i segnali analogici controllati dai calcolatori con circuiti simili a quello qui illustrato nella Figura 1-1; essi possono essere utilizzati per:

- Pilotare un servo-motore
- Pilotare un registratore su carte
- Controllare un convertitore tensione-frequenza
- Controllare un alimentatore programmabile
- Pilotare un rivelatore analogico

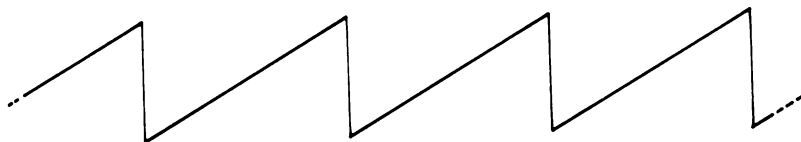
In alcune applicazioni è necessario generare una rampa ad incremento lineare che permetta il controllo dell'andamento di segnali, di muovere una penna di un registratore o

una traccia di oscilloscopio. Nell'esempio seguente useremo un convertitore D/A per generare un segnale adatto a questo scopo.

GENERAZIONE DI UNA RAMPA

La rampa di tensione generata da un convertitore D/A è probabilmente il segnale più semplice che può essere generato con l'uso di un programma relativamente poco complesso. Anche se la vostra conoscenza si limita ad un numero ristretto di istruzioni del microprocessore 8080, probabilmente siete in grado di elaborare un metodo per ideare la sequenza incrementale di numeri binari necessari al convertitore D/A. In questo esempio si suppone l'uso del convertitore interfacciato con il microcalcolatore così come illustrato nella Figura 1-1. Un'istruzione OUT 027 trasferisce il contenuto del registro A dell'8080 al circuito latch del convertitore D/A.

Una delle tecniche più semplici che si possa utilizzare è quella di usare l'istruzione di incremento di un registro.

Una rampa tipica è quella qui riportata:

Incrementando il valore di uno dei registri dell'8080 e trasferendolo nel circuito latch del convertitore, è possibile generare i 256 incrementi in circa 3,2 millisecondi (aumentando una velocità di clock di 500 nanosecondi). La durata di ogni rampa e, di conseguenza, la pendenza, può essere allungata introducendo una routine di ritardo nel programma di generazione della rampa stessa.
Nel programma dell'Esempio 1-1 è illustrata la sequenza di istruzioni necessaria per la generazione della rampa, mentre nell'Esempio 1-2 è riportato il programma con l'inserimento di un ritardo usando una chiamata a sottoprogramma. Il sottoprogramma di ritardo può essere una serie di istruzioni "no-operation" (NOP) oppure qualcosa di più complesso e programmabile in modo da soddisfare diverse esigenze. In questo secondo esempio, è stata scritta e assemblata una routine di ritardo ad impiego generale.

ESEMPIO 1-1
PROGRAMMA TIPICO PER UNA RAMPA POSITIVA

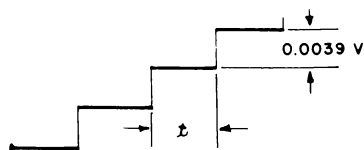
```
                        * 003 000
003 000 074    STAR,    INTRA     ;INCREMENTA IL CONTENUTO DI A
003 001 323             OUT        PORLO IN USCITA AL DAC
003 002 027             027        CODICE DISPOSITIVO DEL DAC
003 003 303             JMP        ESEGUIRLO IN CONTINUAZIONE
003 004 000             START
003 005 003             0
```

ESEMPIO 1-2
PROGRAMMA PER UNA RAMPA POSITIVA CON RITARDO

```
                       * 003 000
003 000 061            LIXISP     CARICA UN INDIRIZZO A STACK
003 001 377            377        PER POTER USARE DELLA SUBROUTINE
003 002 003            003
003 003 074   LOOP,    INTRA      INCREMENTA A
003 004 323            OUT        PORLO IN USCITA VERSO IL DAC
003 005 027            027
003 006 315            CALL       CHIAMA LA SUBROUTINE DI RITARDO
003 007 014            DELAY
003 010 003            0
003 011 303            JMP        ESEGUILO NUOVAMENTE
003 012 003            LOOP
003 013 003            0
```

QUESTA È LA SUBROUTINE DI DELAY

```
003 014 365   DELAY,   PUSHPSW    SALVATAGGIO DEI REGISTRI E DEI FLAG
003 015 325            PUSHD      SALVATAGGIO DI D E DI E
003 016 021            LXID       CARICAMENTO DEI BYTE DI TIMING
003 017 065            065        065 IN E
003 020 001            001        001 IN D
003 021 033   DEC,     DCXD       DECREMENTA LA COPPIA DE
003 022 172            MOVAD
003 023 263            ORAE       SE L'OR DI D ED E È ≠ 0
003 024 302            JNZ        DECREMENTA ANCORA LA COPPIA
003 025 021            DEC
003 026 003            0
003 027 321            POPD       RIPRISTINA I VALORI DEI REGISTRI
003 030 361            POPPSW     RIPRISTINA A E I FLAG
003 031 311            RET        RITORNO
```

È importante ricordare che il convertititore D/A genera una serie di intervalli discreti di 0,0039 volt, cosicché se espandiamo in realtà la rampa lineare, essa appare così:

Il tempo *t*, dipende dalla velocità con la quale si vuole trasferire l'informazione analogica del calcolatore. Il sottoprogramma di ritardo, illustrato nell'Esempio 1-2, permette di incrementare il tempo *t* di alcuni secondi. Un'operazione di filtro e di integrazione può esser necessaria se occorre una linearità più precisa e non è accettabile l'entità dell'incremento. Nella maggior parte delle applicazioni, il segnale può rimanere filtrato anche se può essere in alcuni casi amplificato per modificare il campo di variazione da 0 a 10 volt, oppure da 0 a 2,35 volt.

In alcune applicazioni può essere necessaria una rampa negativa che decresca da potenziali più alti a potenziali inferiori. Ciò è possibile tramite una soluzione hardware, aggiungendo un invertitore, un amplificatore con guadagno unitario ed alcuni altri componenti all'interfaccia del convertitore D/A. Nel nostro caso è tuttavia più ragionevole cambiare il programma in modo da decrementare il registro in modo da ottenere un andamento decrescente nel valore di uscita del convertitore D/A, adattando di conseguenza una soluzione di tipo software. Il programma di generazione della rampa negativa è riportato nell'Esempio 1-3.

ESEMPIO 1-3
TIPICO PROGRAMMA PER UNA RAMPA NEGATIVA

```
                         * 003 000
003 000 075    STAR,     DCRA       DECREMENTA IL CONTENUTO DI A
003 001 323              OUT        TRASFERISCILO AL DAC
003 002 027              027
003 003 303              JMP        RIPETI
003 004 000              START
003 005 003              0
```

Naturalmente può essere aggiunta la chiamata al sottoprogramma di ritardo, anche se non è stato fatto nell'esempio.

Noterete che quando uno dei registri interni ad otto bit dell'8080 contiene il valore 377 ottale, o 11111111 binario, e viene incrementato di uno, essa diventa 000; così il valore massimo diventa 000; così il valore massimo diventa nuovamente il minimo in modo da iniziare la rampa successiva. Analogamente a quando il valore 000 viene decrementato diventa 377, fornendo la rampa decrescente. In questo modo è necessario reinizializzare i registri ad ogni inizio di rampa.

RAMPE COMPLESSE

Possono verificarsi casi in cui non sia necessaria una intera rampa da un valore minimo ad un valore massimo. Supponiamo, per esempio, di dover interfacciare un convertitore D/A con il nostro calcolatore e che l'uscita dal calcolatore vari da 0 a un fondo scala di 10 volt. La nostra applicazione richiede che l'uscita vari continuamente da un valore minimo ad uno massimo. In questo esempio fissiamo il valore inferiore a 2 volt e quello superiore a 8 volt.
Utilizzando la completa gamma da 0 a 10 volt, dovremo impiegare degli amplificatori operazionali per attenuare e limitare il campo di variazioni. Adotteremo invece un accorgimento di tipo software.

Si tratta di un tipico "hardware/software tradeoff".
Nel caso in cui si adotti uno schema con un amplificatore operazionale, si aggiunge dell'hardware per modificare i valori limite, sia superiore che inferiore, cosicché diventa più difficile la modifica che non nel caso di una soluzione software. La soluzione hardware ha tuttavia il merito di mantenere i 256 passi, con incrementi più piccoli, compresi nel nuovo campo di variazione ottenuto attraverso l'aggiunta dell'amplificatore operazionale. Ciò significa che, mentre nella soluzione software, si avranno comunque incrementi di 0,0039 volt, nel caso di conversione da 0 a 10 volt, nel nostro attuale esempio, con una soluzione di tipo hardware, il campo da 2 a 8 volt viene diviso in 256 passi di valore unitario inferiore.

Il primo passo nella risoluzione di questo problema è di determinare i valori binari delle tensioni del limite massimo e limite minimo. Questo calcolo può essere eseguito empiricamente montando un convertitore D/A e incrementando gli ingressi binari e rilevando i valori in uscita da 2 a 8 volt. I valori possono essere calcolati anche così:

$$\frac{\text{10 volt di fondo scala}}{\text{256 passi}} = \text{0,0390 volt/passo}$$

e:

$$\frac{\text{2,0 volt}}{\text{0,0390 volt/passo}} = \text{51,3 passi}$$

Naturalmente è impossibile ottenere frazioni di incremento. Gli unici valori ammessi sono tutti i numeri interi compresi tra 0 e 225. Si deve allora decidere se scegliere come valore limite 51 passi (1,989 volt) o 52 (2,028 volt). In questo caso il valore limite inferiore. In un modo del tutto simile si può stabilire che il valore di limite superiore è rappresentato da 205 passi.

I valori riportati nella tabella seguente rappresentano i valori di limite superiore e inferiore e il loro corrispondente valore binario, ottale o esadecimale.
Questa conversione è necessaria in modo da rendere i valori utilizzabili nei programmi dell'8080.

TENSIONE	PASSI	VALORE BINARIO	OTTALE	ESADECIMALE
2,0 (1, 989)	51	00110011	063	33
8,0 (7, 995)	205	11001101	315	CD

Il programma riportato nell'esempio 1-4 parte dal limite inferiore e incrementa l'uscita dal convertitore D/A fino al limite superiore di 8,0 volt, con incrementi discreti di 39 millivolt.

ESEMPIO 1-4
RAMPA IN USCITA CON LIMITI SUPERIORE E INFERIORE

```
                          * 003 000
003 000 076    START,     MVIA       CARICA IL REGISTRO A CON IL VALORE INIZIALE
003 001 063               063        063 = 051 DECIMALE = 2,0 VOLT
003 002 323    LOOP,      OUT        PONI IN USCITA IL VALORE SUL DAC
003 003 027               027
003 004 074               INRA       INCREMENTA IL VALORE
003 005 000               NOP        TRE ISTRUZIONI NOP PER EVENTUALE
003 006 000               NOP        AGGIUNTA DI UN'ISTRUZIONE DI CHIAMATA AL
                                     PROGRAMMA DI RITARDO
003 007 000               NOP
003 010 376               CPI        ESEGUI IL CONFRONTO DEL VALORE DI LIMITE
003 011 315               315        315 = 205 DECIMALE = 8,0 VOLT
003 012 302               JNZ        SE NON È UGUALE, ESEGUI DI NUOVO IL CICLO
003 013 002               LOOP
003 014 003               0
003 015 303               JMP        SE UGUALE, REINIZIALIZZA IL VALORE INIZIALE
003 016 000               START      ED ESEGUI DI NUOVO
003 017 003               0
```

Il programma dell'Esempio 1-4 causa un'uscita massima sul convertitore D/A di 39 millivolt inferiore al valore massimo.
Ciò avviene quando la comparazione tra il valore limite e il valore corrente del registro è eseguita dopo l'incremento, ma prima che il valore venga caricato nel convertitore. Nel caso in cui si voglia raggiungere il valore effettivo, si dovrà cambiare il valore da confrontare da 315 a 316.

Questo semplice programma può essere utilizzato per realizzare rampe tra due valori qualsiasi all'interno del campo di variazione dell'uscita del convertitore D/A. La rampa può anche essere realizzata con pendenza negativa usando l'istruzione di decremento, confrontando il valore corrente del registro con il valore di limite inferiore e, una volta raggiunto, iniziare dal valore superiore.

La pendenza del segnale in uscita è determinata dal tempo richiesto dal calcolatore per eseguire tutte le istruzioni. Tre istruzioni NOP sono state previste nell'Esempio 1-4, in modo da poter inserire una chiamata ad un sottoprogramma di ritardo. Per un'ottimizzazione della velocità, i NOP possono essere tolti e il programma compattato. Il sottoprogramma di ritardo può essere sia un programma relativamente complesso, come un semplice ciclo di attesa, un flag, un interrupt o altri avvenimenti possono determinare l'esecuzione o meno del passo di programma successivo. Si può quindi aggiungere del software in modo da condizionare l'esecuzione di una nuova rampa al verificarsi di un evento esterno.

FORME D'ONDA TRIANGOLARI

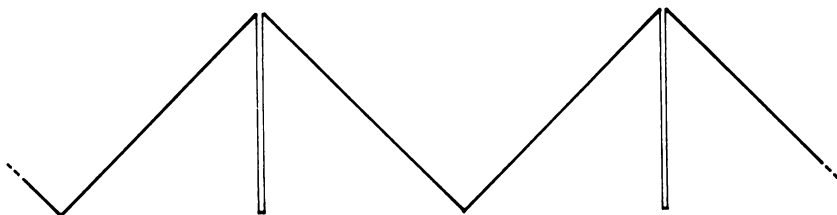
Il precedente esempio per la generazione di una rampa potrebbe servire come base per un programma che permetta la generazione di una forma d'onda triangolare in uscita al convertitore D/A. Nell'esempio attuale si propone di realizzare una forma d'onda triangolare che utilizzi l'intero fondo scala da 0 a 10 volt. Il primo esempio di programma che realizza questo tipo di forma d'onda è illustrato nell'Esempio 1-5.

ESEMPIO 1-5
PROGRAMMA N.1 PER LA GENERAZIONE DI UNA FORMA D'ONDA TRIANGOLARE

```
                        * 003 000
003 000 257             XRAA      AZZERA IL REGISTRO A
003 001 074   UP,       INRA      INCREMENTA IL REGISTRO A
003 002 323             OUT       TRASFERISCE IL CONTENUTO DI A
003 003 027             027
003 004 302             JNZ       SE ANCORA NON È ZERO, CONTINUA
003 005 001             UP
003 006 003             0
003 007 075   DOWN,     DCRA      DECREMENTA IL REGISTRO A
003 010 323             OUT       TRASFERISCE IL CONTENUTO DI A AL
                                  CONVERTITORE
003 011 027             027
003 012 302             JNZ       È ZERO?
003 013 007             DOWN      NO, DECREMENTA NUOVAMENTE
003 014 003             0
003 015 303             JMP       SI, INCOMINCIA DA CAPO
003 016 001             UP
003 017 003             0
```

Se si fa eseguire questo programma al microprocessore, si noterà un "glitch" all'apice del triangolo; ciò è causato dal programma. Infatti dopo l'incremento del registro contatore al suo valore massimo di 377 il programma esegue un nuovo incremento al valore 000 ed

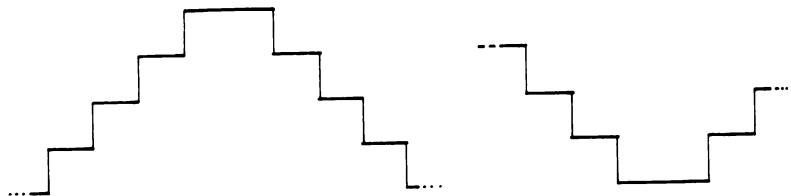
esegue un'operazione di caricamento sul convertitore prima di effettuare l'istruzione di test JNZ.

Quando il valore 000 viene raggiunto, l'istruzione JNZ viene "ignorata" e inizia l'esecuzione della parte DOWN. L'illustrazione indica ciò che si osserverebbe all'uscita dal convertitore durante l'esecuzione del programma dell'Esempio 1-5, assumendo un comportamento ideale del convertitore.

Il glitch può essere eliminato con una semplice inversione di posizione tra le istruzioni di INRA e OUT 027. Il programma modificato, riportato come Esempio 1-6, produrrà una corretta forma d'onda triangolare in uscita. Questo nuovo programma, Programma N. 2, esegue il confronto del valore incrementato, prima dell'istruzione di OUT, evitando la sequenza 377, 000, 377 generata dal primo programma dell'Esempio 1-5.

ESEMPIO 1-6
PROGRAMMA N.2 PER LA GENERAZIONE DI UNA FORMA D'ONDA TRIANGOLARE

```
                       * 003 000
003 000 257            XRAA      AZZERA IL REGISTRO A
003 001 323   UP,      OUT       TRASFERISCE IL VALORE DEL REGISTRO A
                                 AL DAC
003 002 027            027
003 003 074            INRA      INCREMENTA IL REGISTRO A
003 004 302            JNZ       SE NON È ZERO, VA NUOVAMENTE A UP
003 005 001            UP
003 006 003            0
003 007 075   DOWN,    DCRA      DECREMENTA IL REGISTRO A
003 010 323            OUT       TRASFERISCE IL CONTENUTO DI A AL DAC
003 011 027            027
003 012 302            JNZ       SE NON È ZERO, VA NUOVAMENTE AL DOWN
003 013 007            DOWN
003 014 003            0
003 015 303            JMP       SE È ZERO VA AD UP
003 016 001            UP
003 017 003            0
```

Esiste tuttavia un altro problema con questo secondo programma. Sia il valore massimo 377, che il valore minimo 000, infatti, vengono generati due volte consecutivamente, fornendo due appiattimenti della forma d'onda triangolare sia nel suo punto più alto che

in quello più basso. Questi appiattimenti hanno una lunghezza doppia rispetto agli altri gradini generati dal programma, come indica chiaramente il disegno.

Anche questo problema è dovuto a come viene realizzato il programma. Dopo il valore 377 viene messo in uscita sul convertitore D/A, tramite l'istruzione OUT 027 nel ciclo UP, l'accumulatore (registro A) viene incrementato al valore 000, riconosciuto dall'istruzione JNZ e nuovamente al valore 377. A questo punto il valore viene nuovamente messo in uscita durante l'esecuzione del ciclo DOWN del programma, realizzando una ripetizione del valore stesso.
È evidente che alcune modifiche possono essere effettuate nel programma per correggere questo comportamento.

Al programma illustrato nell'Esempio 1-7, vengono aggiunte delle opportune istruzioni di incremento e decremento per eliminare la doppia uscita dei valori 377 e 000.
Questo programma fornisce in uscita segnali a forma d'onda triangolare privi di "glitch".
Va comunque osservato che probabilmente esiste più di una soluzione che risolve questo problema.

ESEMPIO 1-7
PROGRAMMA N.3 PER LA GENERAZIONE DI UNA FORMA D'ONDA TRIANGOLARE

```
                          * 003 000
003 000 323   UP,         OUT       TRASFERISCE IL VALORE DAL REGISTRO A AL
                                    DAC
003 001 027               027
003 002 074               INRA      INCREMENTA IL REGISTRO A
003 003 302               JNZ       SE NON È ZERO, VA DI NUOVO AD UP
003 004 000               UP
003 005 003               0
003 006 075               DCRA      SE È ZERO, LO DECREMENTA AL VALORE 377
003 007 075   DOWN,       DCRA      DECREMENTA IL REGISTRO A NUOVAMENTE
003 010 323               OUT       LO TRASFERISCE AL DAC
003 011 027               027
003 012 302               JNZ       SE NON È ZERO/ VA DI NUOVO AL DOWN
003 013 007               DOWN
003 014 003               0
003 015 074               INRA      SE È ZERO, LO INCREMENTA AL 001
003 016 303               JMP       RITORNA NUOVAMENTA A UP
003 017 000               UP
003 020 003               0
```

La forma d'onda triangolare generata dal programma dell'Esempio 1-7 ha le rampe positiva e negativa della stessa durata dal momento che le due parti del programma UP e

DOWN contengono lo stesso numero di istruzioni e con uguali tempi di esecuzione. Nel caso in cui si vogliono generare forme d'onda in cui le due rampe siano differenti, basterà semplicemente aggiungere alcune istruzioni NOP oppure delle chiamate a sottoprogrammi di ritardo. Per esempio, se al ciclo DOWN, tra le istruzioni di OUT 027 e JNZ, vengono aggiunte 11 istruzioni NOP, il rapporto tra la rampa positiva e quella negativa sarà di 3 a 7.

ONDE QUADRE E ALTRI TIPI DI SEGNALI

Le onde quadre sono facilmente generabili con l'uso di un convertitore D/A e un programma che generi alternativamente valori di tensione alti e bassi. I periodi e le frequenze possono essere determinati sia con le istruzioni NOP che con le chiamate a sottoprogrammi di ritardo. Un tipico esempio di onda quadra è quello illustrato nell'esempio che segue.

ESEMPIO 1-8
PROGRAMMA PER LA GENERAZIONE DI UN'ONDA QUADRA

```
                         * 003 000
003 000 076   START,     MVIA     CARICA IL REGISTRO A CON IL LIMITE
                                  INFERIORE
003 001 023              023
003 002 323              OUT      TRASFERISCI IL VALORE AL DAC
003 003 027              027
003 004 315              CALL     ESEGUI UNA ROUTINE DI RITARDO
003 005 021              DELAY
003 006 003              0
003 007 076              MVIA     CARICA IL REGISTRO A CON IL LIMITE
                                  SUPERIORE
003 010 307              307
003 011 323              OUT      TRASFERISCI IL VALORE AL DAC
003 012 027              027
003 013 315              CALL     ESEGUI UNA ROUTINE DI RITARDO
003 014 021              DELAY    SIMILE A QUELLA DELL'ESEMPIO 1-2
003 015 003              0
003 016 303              JMP      RICOMINCIA NUOVAMENTE
003 017 000              START
003 020 003              0

003 021 000   DELAY,     0        IL SOTTOPROGRAMMA DI RITARDO INIZIA QUI
```

Modificando il software per utilizzare due sottoprogrammi di ritardo, possono essere generate delle forme d'onda rettangolari. Con un breve ritardo sono generabili degli impulsi, e con programmi con ritardi più consistenti, sono realizzabili i ritardi tra i vari impulsi. In funzione della frequenza del clock dell'8080, sono generabili impulsi molto brevi. Il programma dell'Esempio 1-9 mostra la sequenza di istruzioni da usarsi per generare un impulso di 20 microsecondi; a 10 V., con una frequenza di ripetizione di 10 millisecondi.

ESEMPIO 1-9
PROGRAMMA PER LA GENERAZIONE DI UN TIPICO IMPULSO CON L'USO DI UN DAC

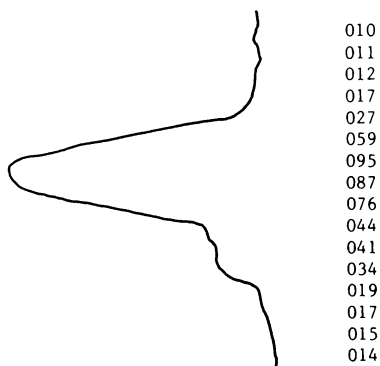
```
                            * 003 000
003 000 076   LOOP,    MVIA        CARICA IL REGISTRO A CON IL VALORE
                                   SUPERIORRE
003 001 377            377
003 002 323            OUT         TRASFERISCILO AL DAC
003 003 027            027
003 004 076   MVIA,    CARICA IL REGISTRO A CON IL VALORE
                                   INFERIORE
003 005 000            000
003 006 323            OUT         TRASFERISCILO AL DAC
003 007 027            027
003 010 315            CALL        ESEGUI LA ROUTINE DI RITARDO
003 011 016            DELAY
003 012 003            0
003 013 303            JMP         RITORNA ALL'INIZIO ED ESEGUI DI NUOVO
003 014 000            LOOP
003 015 003            0

003 016 000   DELAY,   0           UN PROGRAMMA DI RITARDO DI 10
003 017 000            0           MILLISECONDI PUO' ESSERE INSERITO IN
                                   QUESTA POSIZIONE, COME MOSTRATO
                                   NELL'ESEMPIO 1-2.
```

Forme d'onda complesse possono essere prodotte utilizzando i programmi di generazione d'onda quadra, onde triangolari e rampe come sottoprogrammi. Combinando insieme questi sottoprogrammi con una serie di istruzioni di chiamata e di routine di ritardo si possono ottenere uscite complesse. Questi segnali sono generalmente difficili da generare con le apparecchiature standard e possono essere utili per il collaudo di alimentatori, dispositivi elettromedicali, ecc. Mentre bisogna sottolineare la lentezza di variazione dell'uscita in quanto limitata dalla velocità di esecuzione delle istruzioni dell'8080, rimane difficoltoso generare queste complesse forme d'onda in altri modi.

VISUALIZZAZIONE DI DATI IN USCITA

Molte applicazioni richiedono che il calcolatore venga utilizzato per generare "file" di dati che vengono in seguito ulteriormente elaborati. I dati normalmente possono essere forniti in due formati: o una serie di valori numerici stampati su carta o una rappresentazione grafica come un diagramma, un disegno ecc. In molti casi sarà molto più facile da distinguere l'andamento e il cambiamento dei valori che sono stati memorizzati nel calcolatore se si utilizza un'uscita di tipo grafico come ad esempio un oscilloscopio, un plotter X-Y o registratore su carta.
Il disegno che segue mostra la differenza di comprensione tra un'uscita di tipo numerico e una di tipo grafico.

Il prossimo argomento riguarda appunto lo sviluppo di un programma che fornisce in uscita alcuni dati in forma grafica.
Un "file" di parole a otto bit (byte) viene caricato in memoria a partire dall'indirizzo 007 000 fino a 007 143 compresi, in modo da fornirci 100_{10} valori da visualizzare in qualche modo. Avremo a disposizione un registratore su carta compatibile con i valori in uscita dal convertitore D/A che abbiamo precedentemente interfacciato con il nostro sistema 8080.

I valori vengono messi in uscita una volta ciascuno e in considerazione del fatto che il calcolatore è in grado di mettere in uscita tutti e 100 i punti molto più velocemente del tempo di risposta del servo-motore dal registratore, dovremo inserire un ritardo di 100 millisecondi nel programma tra una scrittura in uscita e l'altra.
In questo modo la messa in uscita di tutti i 100 punti richiede 10 secondi.

Per poter lavorare su una serie di dati memorizzati nel sistema 8080, si può utilizzare un puntatore di indirizzo contenuto nei registri H e L, in modo da avere a disposizione i 16 bit necessari. Per puntare alle locazioni di memoria da 007000 a 007143, così come sarà necessario avere a disposizione un sottoprogramma che generi un ritardo di 100 millisecondi.

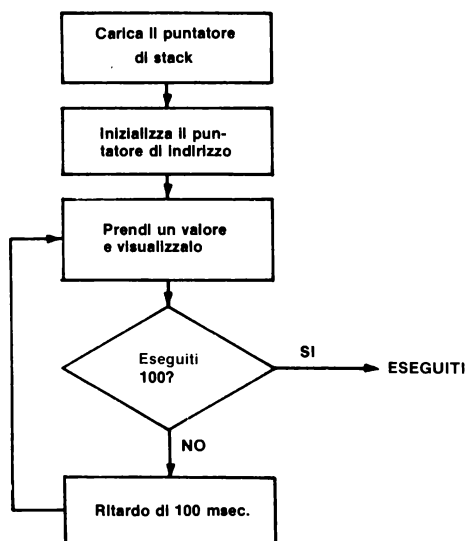



Figura 1-2. Flow di un programma per la visualizzazione o la scrittura su plotter con l'utilizzo di un convertitore D/A a otto bit, di 100 punti.

Sebbene sia importante in alcune applicazioni, ignoreremo il tempo necessario di fatto per prendere il dato dalla memoria e per la sua messa in uscita sul convertitore D/A. Ciò significa che il *"real time"* tra l'emissione di due dati è qualcosa di più di 100 millisecondi, anche se di fatto la differenza è trascurabile. Un registro contatore è inoltre necessario per tenere il conto del numero dei dati emessi in modo da arrestare il programma quando tutti i 100 valori sono stati tracciati. Poiché si farà uso di sottoprogrammi, è necessario predisporre un'area di stack nella memoria RAM ed è buona norma scegliere una zona distante da quella usata per il programma e la memorizzazione dei dati.
Il flow del programma è riportato in Figura 1-2.

Il programma relativo al flow è il seguente:

ESEMPIO 1-10
PROGRAMMA DI INIZIALIZZAZIONE DI 100 PUNTI

		* 003 000	
003 000 061		LXISP	FISSA IL VALORE DI STACK
003 001 377		377	
003 002 003		003	
003 003 041	DISPLA,	LXIH	CARICA LA COPPIA H,2 CON L'INDIRIZZO DI
003 004 000		DATA 1	INIZIO DELL'AREA DI MEMORIA DATI
003 005 007		0	
003 006 006		MVIB	CARICA IL CONTATORE DI PUNTI
003 007 144		144	CON IL VALORE $144_8 = 100$ DECIMALE
003 010 176	LOOP,	MOVAM	PRENDI UN VALORE
003 011 323		OUT	E METTILO IN USCITA
003 012 027		027	
003 013 043		INXH	INCREMENTA L'INDIRIZZO
003 014 005		DCRB	DECREMENTA IL CONTATORE DI PUNTI
003 015 312		JZ	SE TERMINATO, ESCI DAL CICLO
003 016 026		DONE	
003 017 003		0	
003 020 315		CALL	SE NON, ESEGUI IL RITARDO
003 021 027		HUNMIL	DI 100 MILLISECONDI
003 022 003		0	
003 023 303		JMP	VA A PRENDERE IL PROSSIMO PUNTO
003 024 010		LOOP	
003 025 003		0	
003 026 166	DONE,	HLT	STATO DI HLT AL TERMINE DELLA VISUALIZZAZIONE
003 027 000	HUNMIL,	0	IN QUESTO PUNTO SI PUO' INSERIRE LA
003 030 000		0	ROUTINE DI RITARDO. SI VEDA L'ESEMPIO
003 031 000		0	1-2 PER UN TIPICO SOTTOPROGRAMMA DI RITARDO
		* 007 000	
007 000 000	DATA 1,	0	INDIRIZZO DI PARTENZA DEI DATI PER I PROSSIMI 100 INDIRIZZI.

Questo programma pone in uscita l'intero file di 100 dati in circa 10 secondi, o 0,1 secondi per punto. Il registro B del 8080 o qualsiasi altro registro non usato può svolgere la funzione di conteggio dei 100 punti, ma è proprio necessario?
Certamente è necessario un contatore, ma il registro L è gia usato con questa funzione dal momento che viene incrementato in ciascun ciclo di uscita: durante l'esecuzione del programma passa da un valore iniziale di 000 a quello finale di 143.
Il registro L può quindi essere usato per due funzioni:

1) per fornire la parte bassa dell'indirizzo in memoria del dato
2) per tenere il conteggio dei punti posti in uscita.

Comparando il valore del registro L con 144_8 possiamo stabilire quando è stato emesso l'intero file. Si ricordi che le istruzioni di confronto dell'8080 non alterano nessuno dei due dati da comparare.

In questo esempio, il contenuto del registro L viene trasferito nel registro A e poi comparato con il byte 144_8 tramite l'istruzione di comparazione immediata CPI. Non è necessario trasferire nuovamente il valore dal registro A al registro L in quanto durante la istruzione di trasferimento i dati vengono *ricopiati.*

Nel programma dell'Esempio 1-11 il registro L è usato sia come contatore che come parte bassa dell'indirizzo, permettendo così di non usare il registro B e di lasciarlo libero per altri impieghi.

ESEMPIO 1-11
PROGRAMMA MIGLIORATO PER LA VISUALIZZAZIONE DI 100 PUNTI, RIPETIAMO SOLO IL "MAIN PROGRAM"

```
                          * 003 000
003 000 061               LXISP
003 001 377               377
003 002 003               003
003 003 041   DISPLA,     LXIH
003 004 000               DATA 1
003 005 007               0
003 006 176   LOOP,       MOVAM
003 007 323               OUT
003 010 027               027
003 011 043               INXH      INCREMENTA L'INDIRIZZO DI MEMORIA
003 012 175               MOVAL     SPOSTA IL CONTENUTO DEL REGISTRO L IN A
003 013 376               CPI       E COMPARATO CON
003 014 144               144       IL VALORE 144₈ = 100 DECIMALE
003 015 312               JZ        SE TUTTI I PUNTI SONO STATI VISUALIZZATI
003 016 026               DONE      TERMINA IL PROGRAMMA
003 017 003               0
003 020 315               CALL
003 021 027               HUNMIL
003 022 003               0
003 023 303               JMP
003 024 006               LOOP
003 025 003               0
```

USO DEI TERMINALI CRT AD ALTA VELOCITA' CON CONVERTITORI D/A

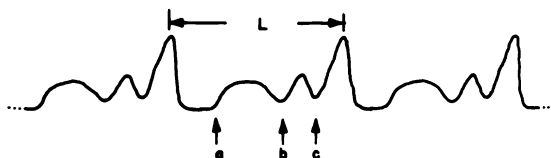
Nell'esempio precedente, l'informazione, costituita da 100 valori corrispondenti a 100 punti, è stata messa in uscita una sola volta ed a bassa velocità. Esistono tuttavia molte altre applicazioni nelle quali si vuole emettere i dati molto velocemente e spesso più di una sola volta. È il caso, ad esempio, di quando si utilizza un CRT (Tubo a Raggi Catodici) o un oscilloscopio.

Eliminando semplicemente i tre byte dell'istruzione di chiamata al sottoprogramma di ritardo, all'interno del programma dell'Esempio 1-11 che emetteva i dati sul registratore su carta, otteniamo l'emissione di 100 punti una sola volta, ad una velocità considerevole, adatta probabilmente per un oscilloscopio. Se utilizziamo tuttavia un oscilloscopio normale (non a memoria) il calcolatore deve emettere i dati continuamente in modo da ottenere una immagine persistente.

Al posto di un salto all'istruzione di indirizzo DONE nel programma per il registratore su carta, (Esempi 1-10 e 1-11) il calcolatore può essere riprogrammato in modo da effettuare una emessione ripetuta continuamente. Ciò si ottiene facendo ritornare il programma al punto in cui avviene il caricamento nei registri H, L dell'indirizzo di inizio dei dati in memoria. Questa soluzione è riportata nell'Esempio 1-12 nel quale è stata eliminata la chiamata al sottoprogramma di ritardo.

ESEMPIO 1-12
PROGRAMMA DI VISUALIZZAZIONE
DI 100 PUNTI PER DISPOSITIVI AD ALTA VELOCITA'

```
                            * 003 000
003 000 061                 LXISP       FISSA IL VALORE DI STACK
003 001 377                 377
003 002 003                 003
003 003 041     DISPLA,     LXIH        CARICA L'INDIRIZZO DI INIZIO DEI DATI
003 004 000                 DATA 1
003 005 007                 0
003 006 176     LOOP,       MOVAM       PRENDI UN VALORE E
003 007 323                 OUT         PONILO IN USCITA SUL DAC
003 010 027                 027
003 011 043                 INXH        INCREMENTA L'INDIRIZZO DI MEMORIA
003 012 175                 MOVAL       SPOSTA IL CONTENUTO DI L IN A
003 013 376                 CPI         E COMPARATO CON 100 DECIMALE
003 014 144                 144
003 015 302                 JNZ         SE NON È UGUALE VA A PRENDERE UN ALTRO
                                        VALORE
003 016 006                 LOOP
003 017 003                 0
003 020 303                 JMP         SE TERMINATI I 100 PUNTI, REINIZIALIZZA
003 021 003                 DISPLA      L'INDIRIZZO ED ESEGUI DI NUOVO
003 022 003                 0
```

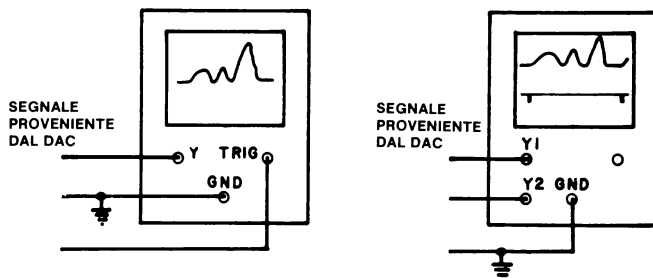
Questo programma emette i 100 punti del file di dati continuamente e la rappresentazione che si ottiene è riportata qui sotto.

Mentre vengono emessi i dati, è difficile stabilire dove comincia il file e dove termina; è possibile invece determinare la lunghezza *L*, dal momento che si verifica la ripetizione del segnale. Tale lunghezza può essere più propriamente chiamata periodo *T*, o ΔT, in quanto è proprio un periodo di tempo che viene misurato dall'oscilloscopio. Il punto più alto non è probabilmente l'inizio del file; è più probabile che esso sia nei punti *a*, *b*, e *c*.

La maggior parte di file di dati presenta normalmente una discontinuità tra l'inizio e la fine, in modo che si possono notare, tuttavia non è da escludere che ciò non si verifichi. In questi casi è necessario trovare un metodo che permetta l'individuazione dell'inizio e della fine del file.

Uno potrebbe essere quello di dare un valore conosciuto al primo elemento del file, per esempio 377, ma ciò spreca uno dei valori del file e può ingenerare confusione nell'interpretazione dei valori stessi. Un modo facile per superare questo problema è di generare tramite il calcolatore un'uscita che possa essere usata come segnale di trigger dell'oscilloscopio-oppure come seconda traccia, qualora sia disponibile, in modo da individuare il punto di inizio.
Il comando di generazione del segnale può essere inserito all'inizio del programma all'etichetta DISPLA come mostrato nell'Esempio 1-13.

ESEMPIO 1-13
GENERAZIONE DELL'IMPULSO DI TRIGGER
NEL PROGRAMMA DI VISUALIZZAZIONE

```
                          * 003 000
003 000 061               LXISP      FISSA IL VALORE DI STACK
003 001 377               377
003 002 003               003
003 003 041    DISPLA,    LXIH       CARICA L'INDIRIZZO DI INIZIO DEI DATI
003 004 000               DATA 1
003 005 007               0
003 006 323               OUT        GENERA L'IMPULSO DI TRIGGER
003 007 030               030        ALL'INIZIO DEL FILE
003 010 176    LOOP,      MOVAM      PRENDE IL VALORE DALLA MEMORIA
```

L'istruzione di OUT 030 genera l'impulso usato come trigger dell'oscilloscopio o per fornire un segnale generico che individua l'inizio del file.

Queste tecniche di visualizzazione di dati si basano sulla base tempi dell'oscilloscopio e sul motore del registratore su carta per generare l'asse dei tempi dei dati in uscita.
Ciò è realizzato tramite il movimento del fascio elettronico dell'oscilloscopio o della striscia di carta ad una velocità di scansione prefissata, ma può essere prodotto con l'impiego di un secondo convertitore D/A che fornisca la base dei tempi o l'asse X.

Il secondo convertitore D/A può quindi essere utilizzato per generare la rampa di tensione necessaria per far partire il fascio elettronico dell'oscilloscopio usando il programma dell'Esempio 1-1. Questo tipo di impiego dei due convertitori per generare dei segnali in uscita controllati dal calcolatore sia per l'informazione che per l'informazione della base dei tempi viene definita visualizzazione di tipo X-Y dal momento che esistono due comandi distinti per ciascun asse.

Lo schema seguente mostra come ciò possa essere realizzato:

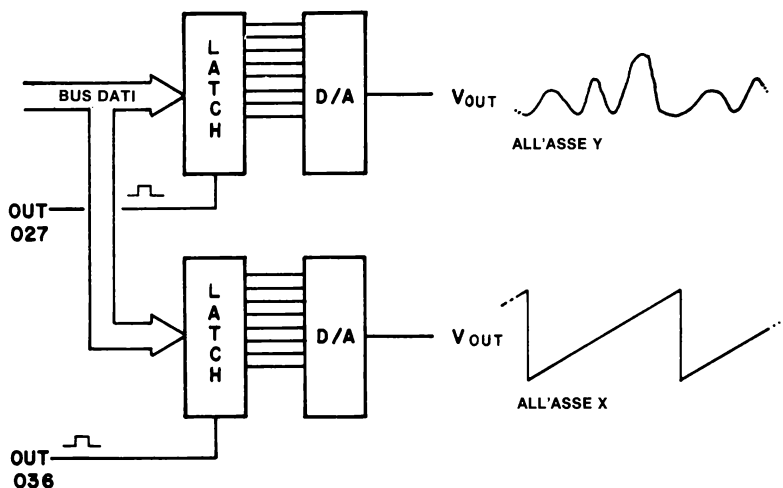



Figura 1-3. Impiego di due convertitori D/A a otto bit per generare tensioni per il controllo degli assi X-Y per una visualizzazione grafica.

Il software deve ora comprendere anche il programma che genera la rampa oltre a quello che emette i dati. Nel caso di visualizzazione di 100 valori o punti, il convertitore D/A che controlla l'asse X o l'asse dei tempi utilizzerà solamente la porzione di 100/256 dell'intero campo di fondo scala dal momento che solo i primi 100 passi di tensione vengono emessi. Questa limitazione si può ovviare utilizzando il controllo di guadagno dell'oscilloscopio per espandere l'asse X fino al riempimento dell'intero schermo. Il programma di controllo per la visualizzazione dei dati nel piano X-Y è riportato nell'Esempio 1-14.

ESEMPIO 1-14
PROGRAMMA PER LA GENERAZIONE DI UNA RAMPA
E DEI DATI UTILIZZANDO L'INTERFACCIAMENTO CON DUE
CONVERTITORI D/A

```
                          * 003 000
003 000 006   DISPLA,   MVIB        INIZIALIZZA IL REGISTRO PER I VALORI DELLA
                                    RAMPA
003 001 000             000
003 002 041             LXIH        INIZIALIZZA L'INDIRIZZO DEI DATI
003 003 000             DATA 1
003 004 007             0
003 005 004   LOOP,     INRB        INCREMENTA IL VALORE DELLA RAMPA
003 006 170             MOVAB       SPOSTA IL VALORE DI RAMPA NEL REGISTRO A
003 007 323             OUT         E LO PONE IN USCITA SUL CONVERTITORE CHE
                                    CONTROLLA L'ASSE X
```

```
003 010 036                036
003 011 176                MOVAM      PRENDE IL PRIMO ELEMENTO DEI DATI
003 012 323                OUT        E LO PONE IN USCITA SUL CONVERTITORE CHE
                                      CONTROLLA L'ASSE Y
003 013 027                027
003 014 043                INXH       INCREMENTA L'INDIRIZZO DI MEMORIA
003 015 175                MOVAL      SPOSTA IL CONTENUTO DEL REGISTRO L IN A
003 016 376                CPI        LO CONFRONTA CON IL VALORE 100 DECIMALE
003 017 144                144
003 020 302                JNZ        SE NON È ZERO, VA A PRENDERE UN ALTRO
                                      PUNTO
003 021 005                LOOP       E LO EMETTE
003 022 003                0
003 023 303                JMP        VISUALIZZATI TUTTI I 100 PUNTI
003 024 000                DISPLA     REINIZIALIZZA E RICOMINCIA DACCAPO
003 025 003                0

                           * 007 000
007 000 000     DATA 1,    0          INDIRIZZO DI PARTENZA DEI DATI
007 001 000                0          PER I PROSSIMI 100 INDIRIZZI.
```

Ogni volta che il calcolatore passa nel ciclo LOOP, il valore del registro B è incrementato e messo in uscita come base dei tempi o asse X. Questo programma può produrre un risultato abbastanza soddisfacente, ma un accurato esame rileverà delle striscie di luce sull'oscilloscopio. Queste striscie sono causate dal breve lasso di tempo per cui ciascun punto è visualizzato in relazione al tempo che impiega l'oscilloscopio nel movimento del fascio elettronico da un punto all'altro.
Può essere quindi necessario introdurre un breve tempo di ritardo nel programma, probabilmente tra le istruzioni di OUT e di INXH, in modo da arrestare e quindi rendere più intensa la luminosità di ciascun punto. Come ritardo si può utilizzare sia un certo numero di istruzioni NOP, sia uno specifico sottoprogramma di ritardo.

Questo programma è interessante dal momento che il valore del registro B e quello del registro L sono sempre uguali. Nel precedente esempio abbiamo visto che il contenuto del registro L può essere usato sia come contantore che come indirizzo di memoria. In questo caso il valore del registro L può essere usato sia come indirizzo di memoria che come valore per il convertitore D/A che controlla l'asse X. Ciò rende nuovamente disponibile il registro B per eventuali altri impieghi.

Anche se l'area dei dati è spostata in modo da avere l'indirizzo di inizio al valore 007 100, il valore nel registro L può ancora essere utilizzato semplicemente sottraendo (o aggiungendo) il valore di offset prima dell'uso del dato come uscita per il convertitore che controlla l'asse X. Questo modo di operare è mostrato nell'Esempio 1-15.

ESEMPIO 1-15
PROGRAMMA PER LA VISUALIZZAZIONE CON CONVERTITORI NEL PIANO X-Y UTILIZZANDO L'INDIRIZZO DI MEMORIA ANCHE COME GENERATORE DI RAMPA

```
                           * 003 000
003 000 061                LXISP      FISSA IL VALORE DI STACK
003 001 377                377
003 002 003                003
003 003 041     DISPLA,    LXIH       INIZIALIZZA IL NUOVO INDIRIZZO DEI DATI
003 004 100                NEWPNT
003 005 007                0
003 006 175     LOOP,      MOVAL      PRENDE LA PARTE BASSA DELL'INDIRIZZO
003 007 376                CPI        E LO CONFRONTA CON IL VALORE 244
                                      OTTENUTO
```

```
003 010 244              244         CON LA SOMMA DI 100 PUNTI (144 OTTALE)
                                     CONL'OFFSET DI 100 (OTTALE) DEGLI INDIRIZZI
003 011 312              JZ          SE UGUALE, RITORNA E RIPETE DA CAPO
003 012 003              DISPLA      SE NON È UGUALE, CONTINUA
003 013 003              0
003 014 326              SUI         SOTTRAETE IL VALORE DI OFFSET
003 015 100              100         DI 100 OTTALE
003 016 323              OUT         IL RISULTATO VIENE UTILIZZATO COME
                                     VALORE
003 017 036              036         DI RAMPA E VIENE EMESSO SUL
                                     CONVERTITORE DELL'ASSE
003 020 176              MOVAM       PRENDE UN VALORE
003 021 323              OUT         E LO EMETTE SUL CONVERTITORE DELL'ASSE Y
003 022 027              027
003 023 315              CALL        ATTENDE UN ATTIMO IN MODO
003 024 032              TIMER       DA PERMETTERE L'INTENSIFICARSI DEL
                                     PUNTO
003 025 003              0
003 026 043              INXH        INCREMENTA L'INDIRIZZO DI MEMORIA
003 027 303              JMP         ED EFFETTUA UNA NUOVA EMISSIONE
003 030 006              LOOP
003 031 003              0

003 032 006   TIMER,     MVIB        CARICA IL REGISTRO B CON UN VALORE
003 033 200              200
003 034 005              DCRB        LO DECREMENTA
003 035 302              JNZ         SE NON È ZERO, RITORNA ALL'ISTRUZIONE
                                     PRECEDENTE
003 036 034              TIMER+2
003 037 003              0
003 040 311              RET         SE IL CONTEGGIO È ZERO, RITORNA AL
                                     PROGRAMMA PRINCIPALE

                         * 007 100
007 100 000   NEWPNT,0               INIZIO DELLA NUOVA AREA DI DATI
```

Probabilmente vi sarete chiesti come mai si è scelto di aggiungere un altro convertitore D/A al microcalcolatore quando si ha a disposizione sia sull'oscilloscopio che sul registratore una base tempi più che valida. La scelta è dovuta al fatto che una rappresentazione sul piano X-Y permette al calcolatore la realizzazione di grafici complessi che non sono rappresentabili da una visualizzazione di tipo Y-T.

Esistono due tipi di visualizzatori X-Y:

1) I dati sono visualizzati utilizzando una base tempi costante con intervalli regolari tra ciascun punto.
2) I dati sono divisi in due "file" che permettono la definizione della posizione dal punto del disegno o del diagramma.

Nel caso di impiego di visualizzatori del secondo tipo , vengono assegnate delle coordinate per ciascun punto dello schermo dell'oscilloscopio in modo da permettere la rappresentazione di onde quadre, cerchi, caratteri alfanumerici e altre forme e figure realizzabili solo con due convertitori D/A.
E' possibile infatti indirizzare 65.536 punti con due convertitori D/A a otto bit, mentre la semplicità di impiego di un visualizzatore Y-T può giustificare il secondo convertitore come generatore di rampa, come é stato illustrato nell'esempio precedente.

Affronteremo ora l'argomento di come impiegare due convertitori D/A per generare in uscita un segnale di tipo X-Y partendo da due file di 100 punti ciascuno. La lunghezza dei

file può essere aumentata o diminuita a seconda della necessità. Se il programma prevede l'impiego di un plotter X-Y, é necessario aggiungere dei segnali di controllo all'interfaccia per il comando del pennino (pennino su-pennino giù) e un ritardo per permettere al servo motore di eseguire il comando del calcolatore. Tutto ciò é stato inserito nel programma. Nel seguito é riportato sia lo schema a blocchi del programma in modo da illustrare il funzionamento che il listing completamente assemblato. Il programma emette i dati continuamente e nel caso di impiego di un plotter deve essere eseguita l'istruzione di HALT.

Nella maggior parte dei casi la coppia di registri 71,2 é usata come puntatore per indicare quale locazione di memoria é implicata nel trasferimento di un dato da o a uno degli altri registri interni dell'8080. Nel programma di emissione dati mostrato nell'Esempio 1-16, la coppia di registri 71,2 é usata come puntatore al file relativo all'asse Y e la coppia di registri D,E come puntatore al file relativo all'asse X. Il file Y é allocato agli indirizzi da 007 000 a 007 177 e il file X da 007 200 a 007 377, impiegando 128 posizioni di memoria ciascuna, anche se nel nostro esempio utilizziamo solo 100 punti.

Nel caso di impiego di un oscilloscopio con il programma dell'Esempio 1-16, la chiamata al sottoprogramma WAIT 1 può essere eliminata sostituendo i tre byte con tre istruzioni NOP. I comandi del pennino possono essere utilizzati, se volete, per attivare o meno il fascio elettronico. Questo impiego prende il nome di *modulazione dell'asse Z.*

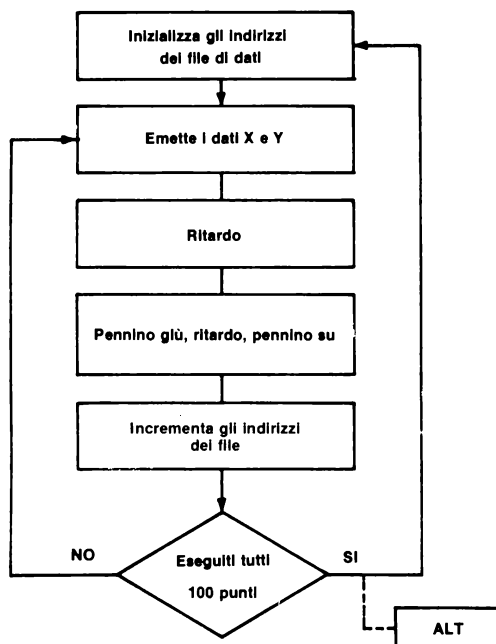



Figura 1-4. Lo schema a blocchi per una routine di rappresentazione sul piano X-Y può essere usato per guidare un plotter X-Y con il comando del pennino in su o in giù. L'istruzione ALT viene utilizzata per l'uscita al plotter con il salto finale da usarsi quando i valori devono essere nuovamente messi in uscita.

ESEMPIO 1-16
PROGRAMMA PER LA VISUALIZZAZIONE DI DATI X-Y CON L'IMPIEGO DI DUE FILE INDIPENDENTI

```
                        * 003 000
003 000 061             LXISP       FISSA IL VALORE DI STACK
003 001 377             377
003 002 003             003
003 003 041   START,    LXIH        INIZIALIZZA IL PUNTATORE PER I DATI
                                    DELL'ASSE Y
003 004 000             Y DATA
003 005 007             0
003 006 021             LXID        INIZIALIZZA IL PUNTATORE PER I DATI
                                    DELL'ASSE X
003 007 200             X DATA
003 010 007             0
003 011 176   MORE,     MOVAM       PRENDE IL VALORE DELL'ASSE Y
003 012 323             OUT         E LO EMETTE
003 013 027             027
003 014 032             LDAXD       PRENDE IL VALORE DELL'ASSE X
003 015 323             OUT         E LO EMETTE
003 016 036             036
003 017 315             CALL        ASPETTA PER IL MOVIMENTO DEL PLOTTER
003 020 100             WAIT1
003 021 003             0
003 022 323             OUT         EMETTE L'IMPULSO DI COMANDO PER IL
                                    PENNINO GIU'
003 023 030             030         OPPURE ATTIVA IL FASCIO DELLO
                                    OSCILLOSCOPIO
003 024 315             CALL        ATTENDE PER LA SCRITTURA O
                                    L'INTENSIFICAZIONE
003 025 130             WAIT2       DEL PUNTO
003 026 003             0
003 027 323             OUT         EMETTE IL COMANDO PER IL PENNINO SU'
003 030 031             031         OPPURE DISATTIVA IL FASCIO DELLO
                                    OSCILLOSCOPIO
003 031 043             INXH        INCREMENTA I PUNTATORI ALLA MEMORIA
003 032 023             INXD
003 033 175             MOVAL       PRENDE LA PARTE BASSA DELL'INDIRIZZO DEI
003 034 376             CPI         DATI DELL'ASSE Y E SI CHIEDE SE HA
                                    TERMINATO
003 035 144             144
003 036 302             JNZ         NO, EMETTE UN ALTRO PUNTO
003 037 011             MORE
003 040 003             0
003 041 303             JMP         SI, RICOMINCIA DI NUOVO
003 042 003             START       QUESTA ISTRUZIONE DEVE ESSERE
                                    SOSTITUITA
003 043 003             0           DA HALT NEL CASO DI UNA SOLA EMISSIONE
                        * 003 100
003 100 000   WAIT1,    0           INSERIRE UN APPROPRIATO
                                    SOTTOPROGRAMMA DI RITARDO

                        * 003 130
003 130 000   WAIT2,    0           INSERIRE UN APPROPRIATO
                                    SOTTOPROGRAMMA DI RITARDO
                        * 007 000
007 000 000   Y DATA,   0           INIZIO DELL'AREA DI DATI DELL'ASSE Y
                        * 007 200
007 200 000   X DATA,   0           INIZIO DELL'AREA DI DATI DELL'ASSE X
```

ESEMPIO 1-17
PROGRAMMA PER LA VISUALIZZAZIONE DI TIPO X-Y
UTILIZZANDO UN SINGOLO FILE SIA PER I DATI
DELL'ASSE X CHE PER I DATI DELL'ASSE Y

```
                          * 003 000
003 000 061               LXISP
003 001 377               377
003 002 003               003
003 003 041     START,    LXIH       CARICA IL PUNTATORE ALLA MEMORIA
003 004 000               XYDATA
003 005 007               0
003 006 176     LOOP,     MOVAM      PRENDE IL VALORE RELATIVO ALL'ASSE Y
003 013 323               OUT        E LO EMETTE
003 010 027               027
003 011 043               INXH       INCREMENTA IL PUNTATORE
003 012 176               MOVAM      PRENDE IL VALORE DELL'ASSE X DALLA
                                     LOCAZIONE SUCCESSIVA
003 013 323               OUT        E LO EMETTE
003 014 036               036
003 015 315               CALL       QUESTA PARTE DI PROGRAMMA E' LA STESSA
003 016 100               WAIT1      DELL'ESEMPIO 1-16
003 017 003               0
003 020 323               OUT
003 021 030               030
003 022 315               CALL
003 023 130               WAIT2
003 024 003               0
003 025 323               OUT
003 026 031               031
003 027 043               INXH       INCREMENTA DI NUOVO IL PUNTATORE
003 030 175               MOVAL      COMPONE IL CONTENUTO DEL REGISTRO L
003 031 376               CPI        CON IL VALORE 2OO DECIMALE
003 032 310               310        310 OTTALE = 200 DECIMALE
003 033 302               JNZ
003 034 006               LOOP
003 035 003               0
003 036 303               JMP
003 037 003               START
003 040 003               0

                          * 007 000
007 000 000     XY DATA,0            INIZIO DELL'AREA DI DATI CON IL PRIMO
                                     VALORE DELL'ASSE Y
007 001 000               0          VALORE DELL'ASSE X
007 002 000               0          VALORE DELL'ASSE Y
007 003 000               0          ETC....
```

Il calcolatore può utilizzare un singolo file di dati nel quale le informazioni relative all'asse X e all'asse Y sono poste in locazioni di memoria alternata, eliminando la necessità di impiego della coppia di registri, D,E come puntatore. L'Esempio 1-17 mostra appunto come si procede in questi casi. Evidentemente la lunghezza di questo singolo file é doppia rispetto a quella dei file impiegati nell'Esempio 1-16, anche se l'area di memoria utilizzata é complessivamente la stessa. Uno degli svantaggi di questa struttura di file é che richiede del software addizionale per ulteriori manipolazioni dei dati del file.

Abbiamo proposto esempi solo di visualizzazione di punti singoli. La visualizzazione di linee é molto più complessa in quanto richiede la "creazione" di ulteriori punti in modo da realizzare la striscia.Routine matematiche vengono normalmente utilizzate per aggiungere questi punti, che servono per riempire lo spazio tra due punti reali, ma la visualizzazione di linee é al di fuori della nostra attuale discussione.

INTERFACCIAMENTO DI UN CONVERTITORE D/A A 10 BIT

Molte applicazioni che impiegano segnali di tipo analogico generati da calcolatori richiedono più risoluzioni che quella di una parte su 256 fornita da un convertitore a 8 bit come quello utilizzato negli esempi mostrati finora. Convertitori a 10, 12 e 14 bit sono attualmente disponibili e possono, anche questi, essere interfacciati a sistemi a microprocessori.

Vi chiederete come sia possibile interfacciare un dispositivo a 10, 12 o 14 bit con un calcolatore a 8 bit. In effetti è più semplice di quanto possa sembrare in un primo tempo. Useremo per l'esempio illustrativo un convertitore D/A a 10 bit con una risoluzione quindi di 1 parte su 1024 che é più che sufficiente nella maggior parte delle applicazioni.

La tecnica che permette l'interfacciamento del convertitore a 10 bit con un calcolatore a 8 bit, é quella di fare in modo che il trasferimento avvenga in un primo tempo per gli otto bit meno significativi e in seguito, con un'altra operazione e un'altra circuiteria, per gli ultimi due bit. Il circuito di come ciò può essere realizzato é mostrato in fig. 1-5.

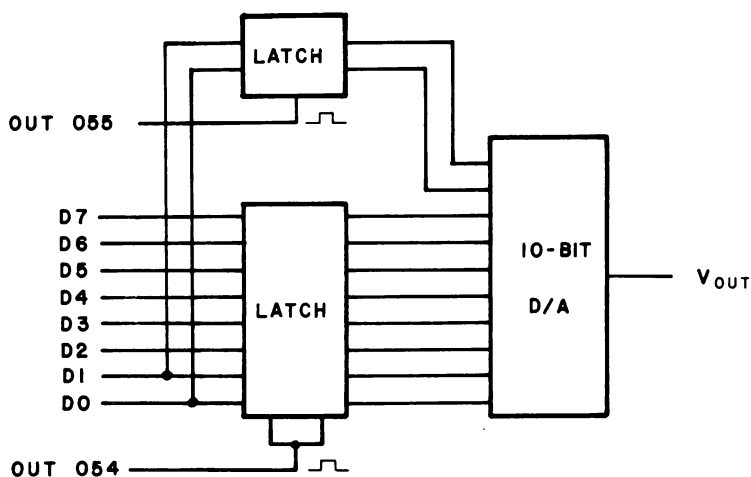



Figura 1-5. Schema del circuito di interfacciamento di un convertitore a 10 bit con un calcolatore a 10 bit. Questa soluzione é approssimativa.

I 10 bit del dato da visualizzare sono memorizzati in due locazioni di memoria consecutiva in questo modo:

Indirizzo	Bit del dato								
A	D7	D6	D5	D4	D3	D2	D1	D0	Parola di 10 bit
A + 1	X	X	X	X	X	X	D9	D8	
A + 2	D7	D6	D5	D4	D3	D2	D1	D0	Parola di 10 bit
A + 3	X	X	X	X	X	X	D9	D8	

X= Porzioni di bit non utilizzata.

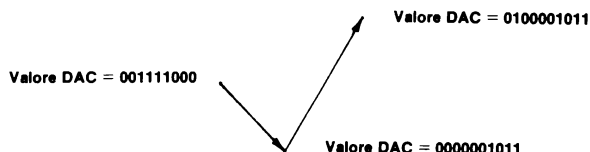
Gli otto bit meno significativi (LSBs) sono memorizzati nella locazione A, mentre i due bit più significativi della parola a 10 bit sono memorizzati nella locazione di memoria successiva A + 1 nelle posizioni di bit D1 e D0. Le altre su posizioni di bit quest'ultima locazione relativa alla parte più significativa (MSB) non sono usate.

Abbiamo supposto l'interfacciamento con un solo convertitore a 10 bit anche se è evidente che è possibile aggiungerne altri. Una parte del programma necessario per comandare il convertitore mostrato in Figura 1-5 è riportato qui di seguito:

```
             * 000 000
000 000 041  LXIH      CARICA IL PUNTATORE DELLA MEMORIA
000 001 000  000
000 002 007  007
000 003 176  MOVAM     PRENDE GLI OTTO BIT MENO SIGNIFICATIVI
000 004 323  OUT       E LI METTI IN USCITA SUL DAC
000 005 054  054
000 006 043  INXH      INCREMENTA L'INDIRIZZO
000 007 176  MOVAM     PRENDE I DUE BIT PIU' SIGNIFICATIVI
000 010 323  OUT       LI METTE IN USCITA SUL DAC
000 011 055  055
```

Supponiamo ora che il convertitore stia emettendo il valore di tensione corrispondente al valore binario di 10 bit 0011110000_2 e che il prossimo valore che il convertitore dovrà generare sia 0100001011_2. Che cosa accadrà se il calcolatore utilizzando la parte di programma mostrato precendetemente, effettuerà questo passaggio?

Ci attendiamo evidentemente che il calcolatore generi, tramite il convertitore, un nuovo valore di tensione che sia superiore al valore corrente, in quanto il secondo valore binario é superiore al primo. Ciò é vero alla fine dell'emissione, mentre a metà del trasferimento si ha un "glitch" nel segnale di uscita.

Il segnale assume un aspetto come quello mostrato qui sotto, dove é stata esagerata la scala dei tempi per evidenziare il fenomeno:

Il motivo per cui si osserva questo glitch inaspettato e quel valore 0000001011_2 associato, é dovuto al fatto che non tutti i 10 bit della nuova parola non sono applicati

contemporaneamente agli ingressi del convertitore D/A. La porzione di programma che é stata usata per aggiornarlo spiega il perché di questo funzionamento:

```
LXIH     DATO PRECEDENTE                            = 0011110000
000
007
MOVAM    PRENDE IL DATO NUOVO
OUT      EMETTE GLI OTTO BIT MENO SIGNIFICATIVI     = 0000001011
054
INXH
MOVAM
OUT      EMETTE I 2 BIT PIU' SIGNIFICATIVI          = 0100001011
055
```

Qui si evidenzia che il valore intermedio 0000001011_2 é generato dalla scrittura degli 8 bit meno significativi del modo dato quando non sono ancora stati aggiornati i due bit meno significativi di quello vecchio. È questo dato intermedio che genera quindi il glitch notato.

Quando, noi vogliamo generare il nuovo dato da 10 bit con la attuale interfaccia che trasferisce solo otto bit per volta, non possiamo evitare che si formi un dato "intermedio" formato parte dal vecchio valore, parte dal nuovo; tutto ciò non si elimina se non modificando sia l'hardware che il software del precedente esempio.

Un latch addizionale a 10 bit può essere aggiunto all'interfaccia per impedire al convertitore D/A di generare il glitch causato dalla sovrapposizione tra il vecchio e il nuovo dato. La tecnica di impiego di due latch viene denominata "doublebuffering" (*doppia bufferizzazione*) ed é riportata nella Figura 1-6. Il circuito latch addizionale può essere abilitato o meno con un comando separato di OUT, ma può anche essere collegato in modo da effettuare il trasferimento dato al momento in cui il calcolatore emette i due bit più significativi del dato stesso. In questo esempio si é scelto di utilizzare un comando separato per controllare il latch a 10 bit.

La sequenza che segue mostra come avviene l'emissione del dato sul convertitore D/A a 10 bit:

1) Gli otto bit meno significativi del nuovo valore vengono trasferiti al latch 74100 collegato al bus dei dati. Ciò avviene tramite l'istruzione OUT 054.
2) I due bit più significativi del nuovo valore vengono trasferiti al latch 7475 collegato al bus dei dati. Ciò avviene tramite l'istruzione OUT 055.
3) L'intera parola di 10 bit trasferita al 74100 e al 7475 é trasferita al latch a 10 bit, ottenuto con un altro 74100 e la seconda metà del 7475, tramite l'istruzione OUT 056.

Questo tipo di interfacciamento con doppia bufferizzazione utilizza lo stesso tipo di programma dell'esempio precendente, con l'eccezione di un comando addizionale di scrittura che fornisce l'impulso necessario per abilitare il nuovo latch a 10 bit.

ESEMPIO 1-18
PROGRAMMA TIPICO PER IL CONTROLLO DI UN DAC A 10 BIT

```
                         * 003 000
003 000 041    START,    LXIH       INIZIALIZZA L'INDIRIZZO DEI DATI
003 001 000              DATA
003 002 007              0
```

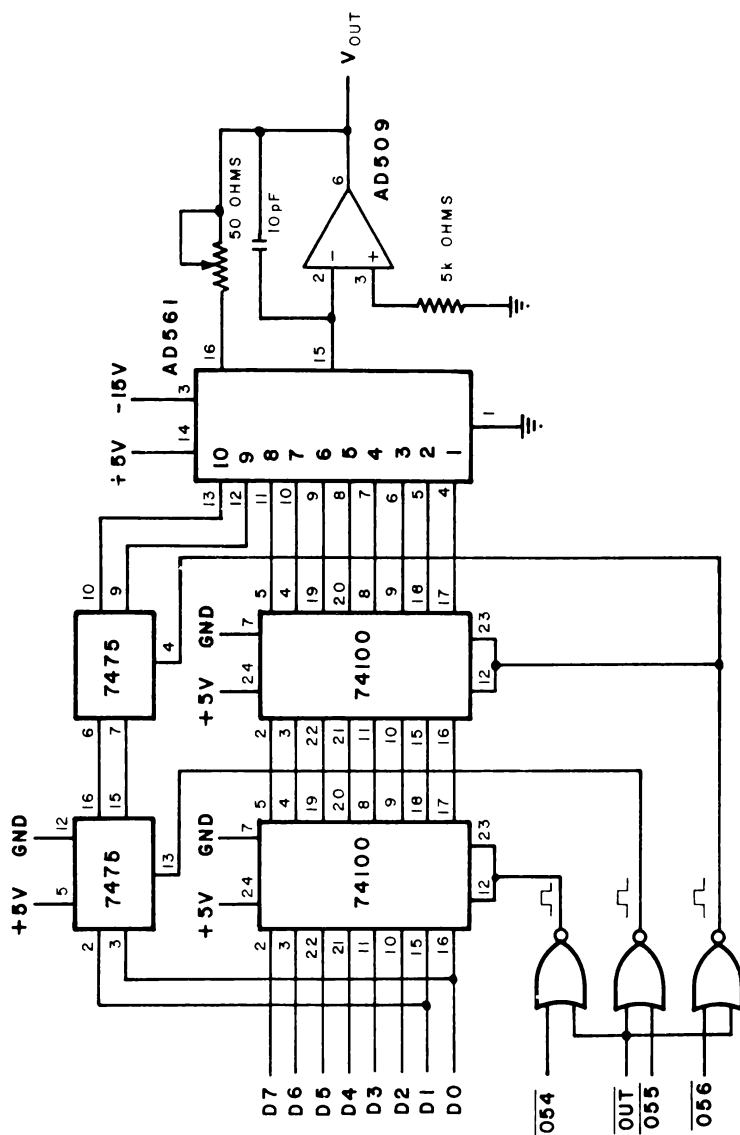



Figura 1-6. Circuito di interfacciamento con il convertitore AD561 dell'Analog Devices a 10 bit con doppia bufferizzazione.

```
003 003 176    DAC,    MOVAM    PRENDE GLI OTTO BIT MENO SIGNIFICATIVI
003 004 323            OUT      E LI EMETTE
003 005 054            054
003 006 043            INXH     INCREMENTA L'INDIRIZZO
003 007 176            MOVAM    PRENDE I DUE BIT PIU' SIGNIFICATIVI
003 010 323            OUT      E LI EMETTE
003 011 055            055
003 012 323            OUT      TRASFERISCE I 10 BIT DEL DATO AL DAC
003 013 056            056      TUTTI IN UNA VOLTA
003 014 043            INXH     INCREMENTA NUOVAMENTE L'INDIRIZZO
003 015 175            MOVAL    PRENDE IL VALORE DEL REGISTRO L
003 016 376            CPI      E SI CHIEDE SE E' = 310 OTTALE
003 017 310            310      310 OTTALE = 200 DECIMALE
003 020 302            JNZ      SE NON HA EMESSO TUTTI E 100 I VALORI
003 021 003            DAC      VA A PRENDERE UN NUOVO DATO
003 022 003            0
003 023 303            JMP      SE SI, REINIZIALIZZA ED ESEGUE
                                NUOVAMENTE
003 024 000            START
003 025 003            0

                       * 007 000
007 000 000    DATA,   0        INDIRIZZO RELATIVO AGLI OTTO BIT MENO
                                SIGNIFICATIVI (DATO 1)
007 001 000            0        INDIRIZZO RELATIVO AI DUE BIT PIU'
                                SIGNIFICATI (DATA 1)
007 002 000            0        INDIRIZZO RELATIVO AGLI OTTO BIT MENO
                                SIGNIFICATIVI (DATO 2)
```

Figura 1-7. Schema a blocchi del convertitore AD7522 dell'Analog Devices a disposizione dei pin. Per gentile concessione dell'Analog Devices Inc.

In uno dei più recenti convertitori D/A, l'AD7522 dell'Analog Devices, il circuito di doppia bufferizzazione é realizzato all'interno del circuito stesso. Per gentile concessione dell'Analog Devices abbiamo riportato lo schema a blocchi del circuito in Figura 1-7.Sono previsti tre ingressi relativi a 3 funzioni: un impulso di trasferimento di otto bit (i meno significativi) LBS al pin 24 uno di trasferimento di due BIT (i più significativi) HBS al pin 25 e un impulso di caricamento del convertitore LDAC al pin 22.

Dal momento che l'AD7522 ha al suo interno la doppia bufferizzazione, abbiamo pensato di utilizzarlo in molti esempi riportati nel Capitolo 7. Per permettere un più semplice interfacciamento il componente é stato montato su una scheda outboard insieme alla circuiteria logica necessaria e al circuito di amplificazione. Lo schema completo per l'outboard di conversione a 10 bit LR-35 é mostrato nalla Figura 1-8, mentre la sua fotografia é illustrata nella Figura 1-9.

La nostra speranza é che i costruttori continuino nella fornitura di moduli sempre più semplici da interfacciare come nel caso del circuito integrato NE 5018 della Signetics. Quest'ultimo é un dispositivo a 22 pin che comprende il circuito di latch, il riferimento analogico e un amplificatore di uscita per un convertitore a 8 bit. L'interfacciamento si riduce a due o tre componenti esterni ed a un dispositivo di generazione dell'impulso di selezione. Questo circuito sarà discusso in maniera più dettagliata negli esperimenti del Capitolo 7.

NOTE:

*ADD Ponticello per 5 Volt
Quando viene utilizzato
il collegamento con il regolatore a 5 Volt, il polo positivo di C1 deve essere collegato direttamente a V REF

**ADD Ponticello per 5 Volt
Quando viene utilizzato ii collegamento con il regolatore a 5 Volt, il polo positivo di C1 deve essere collegato direttamente a massa

Figura 1-8. Schema del circuito dell'Outboard di conversione a 10 Bit LR-35.

Figura 1-9. Fotografia dell'Outboard LR-35. Il convertitore AD 7522 a 10 bit dell'Analog Devices con doppia bufferizzazione é utilizzato per generare un'uscita da 0 a 5 volt.

TECNICHE DI I/O CON MAPPA DELLA MEMORIA E CONVERTITORI D/A

La tecnica di I/O con mappa della memoria può essere facilmente applicata a un convertitore D/A, così come sarà mostrato negli esempi di questo capitolo. Per convertire i circuiti di interfaccia in modo da utilizzare la tecnica di I/O con mappa della memoria al posto della tecnica di I/O tramite accumulatore, i segnali di $\overline{OUT}$ devono essere sostituiti da segnali di scrittura in memoria, $\overline{MW}$ o $\overline{MEMW}$. e l'indirizzo del dispositivo di conversione deve assumere la configurazione di 16 bit piuttosto che quella di 8 degli esempi precedenti. Le tecniche più diffuse di I/O con mappa della memoria sono affrontate nei Capitoli 21 e 22 del Bugbook VI. Gli esempi di programmi devono tuttavia essere modificati in modo che le istruzioni con riferimento alla memoria vengano utilizzate per trasferire i dati al convertitore D/A. Il maggior vantaggio di impiego della tecnica di I/O con mappa della memoria é che si possono utilizzare un vasto numero di istruzioni per trasferire dati tra i dispositivi di I/O e l'8080.

Le interfacce programmabili, come l'8255 della Intel, giustificano l'adozione di tecniche I/O con mappa della memoria, in quanto questi dispositivi mettono a disposizione porte a 8 bit che possono essere programmate sia in ingresso che in uscita.

L'impiego di questo tipo di circuiti integrati che possiedono più di 8 bit per gli ingressi digitali, può comunque creare dei problemi in quanto non possiedono la doppia bufferizzazione, di modo che il convertitore D/A a 10 bit che richiede due circuiti latch non può essere direttamente interfacciato utilizzando un componente del tipo PPI. Se infatti impiegassimo un simile circuito, si verificherebbero quei glitch che derivano dal fatto che i bit all'ingresso non vengono presentati contemporaneamente.

Riteniamo quindi che sia molto meglio utilizzare dei circuiti integrati TTL standard per l'interfacciamento, quando siano ben definite le caratteristiche o comunque sia necessario trasferire più di otto bit in una volta. Comunque rimane la miglior soluzione lo scegliere convertitori D/A con doppia bufferizzazione interna quando si vogliano evitare complicazioni di interfacciamento con convertitori a più di otto bit. Anche da un punto di vista economico, osserverete che esiste poca differenza tra le due soluzioni.

Nelle applicazioni riportate nel Capitolo 7 sono state impiegate tecniche di I/O sia tramite accumulatore che con mappa della memoria e noterete che tutte e due le tecniche risolvono in maniera soddisfacente il problema; la scelta quindi una piuttosto che dell'altra può essere determinata da una vostra abitudine.

ALTRI CONVERTITORI ANALOGICI DIGITALI

Un altro modulo che merita attenzione é il convertitore MP-10 della Burr-Brown. Questo modulo é stato realizzato in funzione di una sua applicazione specifica con microprocessori del tipo 8080, come possono essere l'8085, l'8048, lo Z-80 della Zilog etc.; il suo interesse deriva dal fatto che il modulo è composto al suo interno da due convertitori a 8 bit e da un'interfaccia programmabile (PPI) del tutto simile al componente INTEL 8255. Quest'ultimo é costituito da tre porte che possono essere configurate sia come porte di ingresso che come porte di uscita a seconda di un comando trasferito tramite software che renda il componente completamente programmabile.

L'aver incorporato questo tipo di componente nel package dell'MP-10 ha permesso alla Burr-Brown di mettere a disposizione in un'area di 1 pollice x 1 pollice e 3/4 due convertitori D/A a otto bit che comunicano con il mondo esterno con 32 pin. Uno schema a blocchi del componente é riportato nella Figura 1-10.

Gli ingressi dell'MP-10 sono compatibili TTL, quindi permettono un interfacciamento diretto con i bus degli indirizzi, dei dati e di controllo dei microprocessori. Il dispositivo contiene al suo interno la necessaria logica di decodifica degli indirizzi e può essere configurato in modo da rispondere ai quattro possibili gruppi di indirizzi quando si adotti una tecnica di mappa in memoria.

Ciascun gruppo di indirizzi permette la individuazione dei tre singoli blocchi funzionali presenti all'interno di ciascun dispositivo che corrispondono a:

1° Primo convertitore D/A
2° Secondo convertitore D/A
3° Registro interno di controllo della PPI

Le possibili configurazioni di indirizzi sono:

Parte alta	Parte bassa	Funzione
X X 1 1 1 1 1 1	1 1 1 1 B B 0 0	Convertitore 1
X X 1 1 1 1 1 1	1 1 1 1 B B 0 1	Convertitore 2
X X 1 1 1 1 1 1	1 1 1 1 B B 1 1	Registro di controllo

I bit di indirizzo indicati con la lettera B devono essere determinati dall'utilizzatore tramite opportune connessioni hardware in modo tale da permettere l'impiego in un sistema a microcalcolatore fino a 4 moduli di questo genere.

Gli indirizzi indicati con la lettera B possono assumere configurazioni 00, 01, 10, 11, mentre quelli indicati con la lettera X non vengono considerati all'interno del modulo.

Ogni volta che si utilizza un dispositivo di I/O programmabile in un circuito di interfaccia è necessario inizializzarlo e l'MP-10 della Burr-Brown richiede la stessa procedura che si realizza trasferendo il codice 10000000_2 al registro interno di controllo.
Quest'operazione deve essere eseguita all'inizio dell'uso di uno qualsiasi dei convertitori D/A.

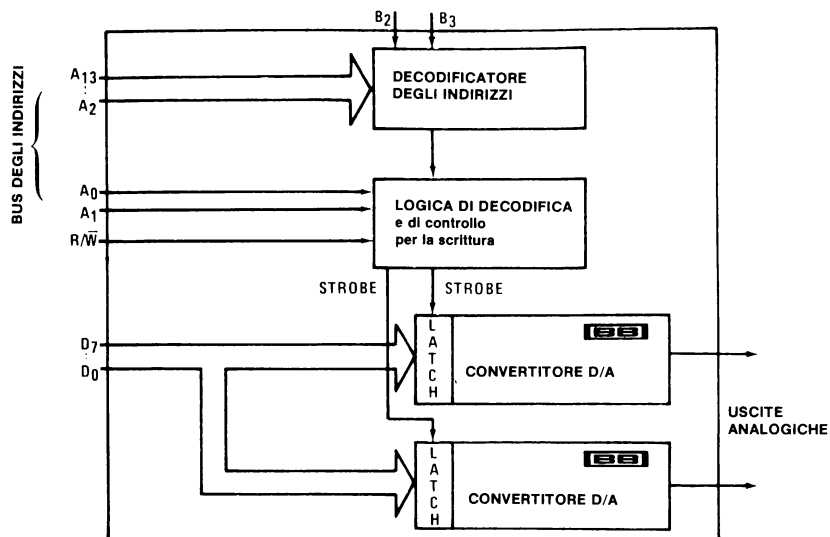



Figura 1-10. Schema a blocchi del modulo a due convertitori D/A a otto bit della Burr-Brown. Per gentile concessione della Burr-Brown Research Corp.

Nel caso di impiego della tecnica di I/O con mappa in memoria, uno dei due esempi riportati nel seguito può essere utilizzato per l'inizializzazione dell'MP-10:

```
MVIA        LXIH
200         CONTROL
STA         0
CONTROL     MVIM
0           200
```

Nell'esempio di sinistra, il registro dell'8080 è caricato con la parola di controllo che deve essere memorizzata nella locazione di memoria di etichette CONTROL che corrisponda all'indirizzo del registro di controllo del convertitore D/A.
Nell'esempio di destra, la coppia di registri H, L è caricato con l'indirizzo del registro di controllo e successivamente la parola di controllo viene trasferita al registro di controllo tramite l'istruzione MVIM. Nei due esempi di programma la notazione CONTROL seguita dal valore 0 viene utilizzata per rappresentare l'indirizzo di 16 bit relativo al registro di controllo del modulo MP-10, cioé $00111111\ 11110011_2$.

Dal momento che ai due convertitori D/A presenti all'interno del modulo MP-10 sono stati assegnati due indirizzi consecutivi è possibile trasferire due valori dei dati, uno per ciascun convertitore tramite l'istruzione SHLD. Questa è un'istruzione a 3 byte che permette il trasferimento del contenuto dei registri H e L nelle locazioni di memoria X e X + 1, dove l'indirizzo X viene specificato negli ultimi due byte successivi dell'istruzione SHLD. In questo modo i due convertitori possono essere aggiornati con il nuovo valore con l'impiego di una sola istruzione.

L'MP-10 non prevede la decodifica dei due bit più significativi A15 e A14, cosicché se si impiega il modulo in una tecnica di I/O con mappa della memoria, esso avrà l'indirizzo

nell'ultimo blocco di memoria di 256 byte per ciascuno dei 4 blocchi di 16 K di memoria disponibili all'interno della possibilità di indirizzamento dell'8080. Ciò è evidenziato dalla seguente tabella di indirizzi:

Decodifica degli indirizzi per il modulo di conversione MP-10 D/A

Indirizzo alto	Indirizzo basso
0 0 1 1 1 1 1 1	1 1 1 1 0 0 0 0
0 1 1 1 1 1 1 1	1 1 1 1 0 0 0 0
1 0 1 1 1 1 1 1	1 1 1 1 0 0 0 0
1 1 1 1 1 1 1 1	1 1 1 1 0 0 0 0

I quattro indirizzi nella tabella indirizzano tutti lo stesso convertitore nel caso in cui gli indirizzi indicati con la lettera B siano stati forzati a zero via hardware. Nel caso in cui si utilizzino 16 K o più di memoria in un unico blocco ciò comporterà dei problemi dal momento che lo stesso indirizzo corrisponderà ad una locazione di memoria e anche a uno dei due convertitori.

Per ovviare a questo inconveniente si può utilizzare lo schema di Figura 1-11 che permette lo spostamento degli indirizzi riguardanti i convertitori e il registro di controllo dell'MP-10 all'ultimo blocco di 256 byte della memoria indirizzabile dall'8080 che è compreso fra gli indirizzi 377 000 e 377 377 (in esadecimale FFOO E FFFF).
La modifica comporta l'impiego di un solo SN7400 che contiene 4 porte NAND.

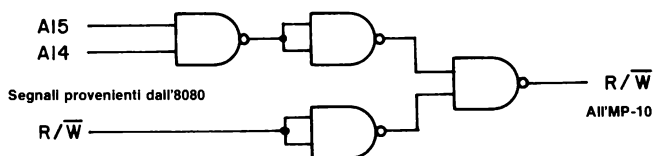



Figura 1-11. Esempio dell'impiego di porte per spostare gli indirizzi del modulo MP-10 ai valori più alti della memoria.

Questo circuito permette l'abilitazione dell'ingresso read/write solo quando tutti e due gli indirizzi A14 e A15 sono a logica 1. Nel caso in cui tutti gli altri bit degli indirizzi, con l'eccezione degli indirizzi da A0 a A3, sono a logica 1, l'indirizzo del dispositivo sarà:

1 1 1 1 1 1 1 1   1 1 1 1 B B C C

Dove le posizioni relative alle lettere "B" devono essere cablate dall'utente e le posizioni delle lettere "C" sono utilizzate per indirizzare le funzioni interne del dispositivo.

Il modulo MP-10 può essere anche impiegato utilizzando la tecnica di I/O tramite l'accumulatore sostituendo il segnale di OUT dell'8080 al segnale di scrittura in memoria da applicare al modulo. In questo caso gli ingressi relativi agli indirizzi A8-A13 devono essere cablati in maniera fissa alla logica 1. Gli indirizzi dei convertitori D/A e del registro di controllo rimangono gli stessi di quelli utilizzati nel caso di impiego della tecnica con mappa della memoria tranne quelli relativi alla parte alta che vengono ignorati. La parte bassa invece viene inserita nel programma come secondo byte dell'istruzione di OUT.

Ricordatevi che nel caso di impiego di tecnica di I/O tramite accumulatore, il registro di controllo interno al modulo MP-10 deve essere inizializzato con il valore 10000000_2.

Sempre nel caso di impiego di I/O tramite accumulatore il programma si modifica nel modo seguente:

```
MVIA        CARICA IL REGISTRO A
200         CON IL VALORE 200
OUT         E LO EMETTE SUL REGISTRO
            DI CONTROLLO
CONTROL     OTTO BIT RELATIVI ALL'INDIRIZZO
            DEL REGISTRO DI CONTROLLO.
```

La Burr-Brown ha realizzato un dispositivo che semplifica l'interfacciamento microcalcolatore-convertitore in quanto ha incorporato all'interno del modulo i latch e la logica di controllo relativa agli indirizzi. Noi ci auguriamo che questa tendenza continui.

È importante comunque sottolineare che l'interfacciamento può essere facilitato, deve comunque esserci il programma che governa il colloquio microcalcolatore-convertitore, scritto in maniera corretta.

CAPITOLO 2

INTERFACCIAMENTO CON CONVERTITORI ANALOGICO-DIGITALI

INTRODUZIONE

I convertitori analogico-digitale assumono un ruolo importante in molti sistemi a microcalcolatore in quanto consentono a un sistema di tipo digitale di convertire un valore di tensione in un valore appunto digitale. Ciò è particolarmente utile quando le grandezze sono temperature, precisioni, pesi, posizioni, distanze o comunque tutte quelle grandezze i cui sensori di misura producono una grandezza (corrente o tensione) di tipo analogico. Il mercato mette a disposizione moduli di conversione A/D in molte configurazioni in grado di risolvere sia problemi specifici che generali.
In questo capitolo presenteremo esempi e applicazioni che impiegano diversi tipi di convertitori e risolvono la problematica di interfacciamento sia da un punto di vista hardware che software.

OBIETTIVI

Alla fine di questo capitolo sarete in grado di:

- Descrivere il funzionamento di un convertitore A/D a rampa.
- Descrivere il funzionamento di un convertitore A/D ad approssimazioni successive.
- Sviluppare i programmi per generare i dati in ingresso sia per convertitori A/D a rampa che ad approssimazioni successive.
- Descrivere l'interfacciamento necessario sia per un modulo a 8 bit, sia per uno a 10 bit.
- Descrivere le differenze tra temporizzatori realizzati con sottoprogrammi, a interruzione e "polled" per programmi di acquisizione dati.
- Discutere l'impiego di convertitori A/D e D/A per applicazioni che richiedono l'acquisizione e la visualizzazione di dati.

CONVERTITORI ANALOGICO-DIGITALE

La maggior parte dei più diffusi convertitori analogico-digitale realizzano la propria funzione confrontando il valore di tensione sconosciuto da misurare con uno prodotto da un convertitore digitale analogico.
La generazione di un valore di tensione conosciuto e l'operazione di confronto sono abbastanza semplici da ritenerle acquisite. Questo capitolo affronterà invece le tecniche di conversione analogico-digitale realizzate tramite programma e i metodi per l'interfacciamento di convertitori modulari A/D con i microcalcolatori.

Convertitori analogico-digitali sono presenti in commercio con un'ampia gamma di configurazioni sia in ingresso o in uscita, come ad esempio buffer di tipo three-state o amplificatori interni al convertitore. I valori di tensione più comuni per cui sono predisposti i convertitori A/D sono 0-5 volt, 0-10 volt, ±5 volt o ±10 volt. Alcuni addirittura sono disponibili con tutti questi valori contemporaneamente o comunque con un amplificatore a guadagno variabile in ingresso in modo da poter misurare qualsiasi valore di tensione. Le uscite dei convertitori A/D sono normalmente compatibili TTL, normalmente codificate in codice BCD; altri codici usuali sono il binario con complemento a due o con "offset" binario.

TECNICHE SOFTWARE A/D

Nell'impiego dei microcalcolatori o di altri piccoli calcolatori in questo campo risulta evidente che le configurazioni sia hardware che software possono essere diverse. Per esempio, le seguenti operazioni possono essere realizzate sia tramite circuiti integrati che istruzioni di programma

- Contatori decimali o binari
- Invertitori
- Shift register
- Conversione di codici
- Flip-flop
- Multiplexer

In effetti, vi sono molti modi di procedere che impiegano soluzioni o hardware o software e nel caso specifico della conversione A/D ambedue le scelte possono essere ugualmente solide.
Analizzeremo ora il funzionamento dei convertitori a rampa e quelli ad approssimazioni successive in modo da studiare come avviene il loro funzionamento e come possa essere completato con del software.

CONVERTITORI A RAMPA LINEARE

Il convertitore a rampa tipico opera confrontando un valore di tensione sconosciuto con una rampa di tensione con incremento lineare. Il confronto è costantemente realizzato fino a quando i due valori non vengono rilevati uguali, in questo caso il processo viene arrestato. Nel caso in cui si utilizzi un contatore per generare la rampa tramite un convertitore D/A, il codice presente agli ingressi del convertitore D/A rappresenterà il codice binario della tensione sconosciuta.

Lo schema della rappresentazione di un convertitore A/D a rampa è mostrato nella Figura 2-1.

Un breve impulso attivo basso è applicato all'ingresso di $\overline{RESET}$ per azzerare sia i contatori che il flip-flop di tipo D. Quando quest'ultimo è azzerato, la sua uscita $\overline{Q}$ è al valore logico 1, permettendo alla porta NAND di far passare gli impulsi ai contatori binari. Il conteggio applicato al convertitore D/A genera la rampa di uscita che tramite il comparatore viene confrontata con il valore di tensione sconosciuto; l'uscita del

comparatore si porrà alla logica 1 non appena i due valori saranno uguali. Quest'ultimo segnale va a livello logico 1, tramite l'ingresso CK al flip-flop; l'uscita $\overline{Q}$ va a 0 bloccando l'arrivo degli impulsi ai contatori.

Quando termina il conteggio, come indicato dal segnale fine conversione segnalato dalla variazione del flag $\overline{DONE}$ che si pone allo stato logico raro, l'uscita parallela di otto bit rappresentarà il codice binario del valore di tensione da conoscere. L'ingresso D del flip-flop è mantenuto alla logica uno al termine del conteggio.

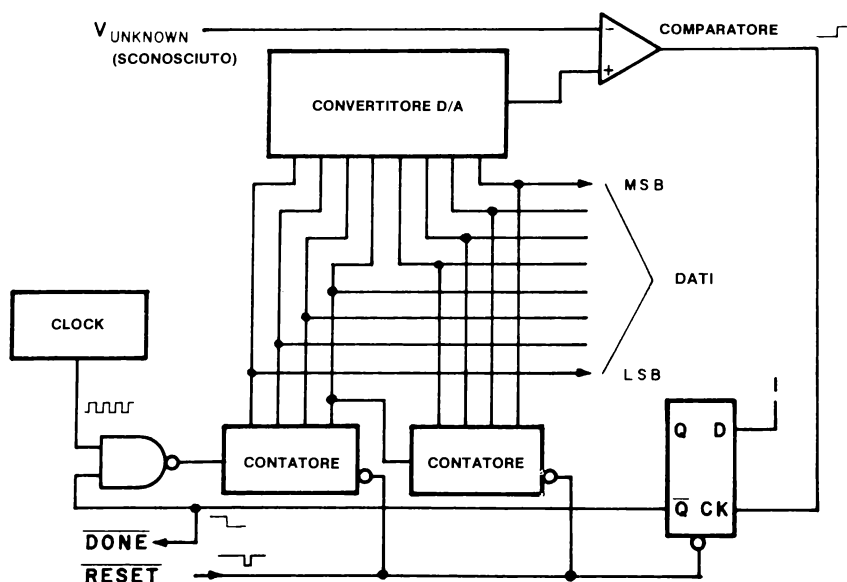



Figura 2-1. Rappresentazione schematica di un tipico convertitore a rampa.

La raffigurazione dell'andamento del fenomeno è riportata nel diagramma dei tempi di Figura 2-2.

Naturalmente si è supposto che la tensione da misurare sia all'interno del campo di variazione di uscita del convertitore a rampa. Nei convertitori di questo tipo è preferibile l'impiego di contatori sincroni piuttosto che quelli asincroni, in quanto le uscite di quelli sincroni cambiano le loro uscite contemporaneamente, evitando quindi impulsi spuri (spikes), generati normalmente da contatori quali un contatore binario a quattro bit 7493. Nel caso occorra un'uscita di tipo BCD è necessario utilizzare un convertitore D/A con ingressi con quel codice oppure dei contatori BCD; nel primo caso il contatore binario 74161 è adatto allo scopo, mentre nel secondo si può prevedere l'impiego del contatore BCD 74160.

Dal momento che abbiamo discusso l'interfacciamento di un convertitore D/A con un microcalcolatore nel precedente capitolo, troveremo modo di utilizzarlo per generare la rampa che ci occorre tramite il programma; il programma provvederà anche ad effettuare il test sull'uscita del comparatore, nel qual caso occorrerà disporre di una porta in ingresso. Lo schema a blocchi dell'interfacciamento è riportato nella figura 2-3.

È possibile a questo punto effettuare una sostituzione di una parte hardware utilizzando una soluzione software, utilizzando i passi di programma presentati nel precedente capitolo per la generazione della rampa in uscita al convertitore D/A. L'uscita del comparatore viene considerato come flag esterno che deve essere continuamente testato dal programma in modo da permettere eventuali decisioni.

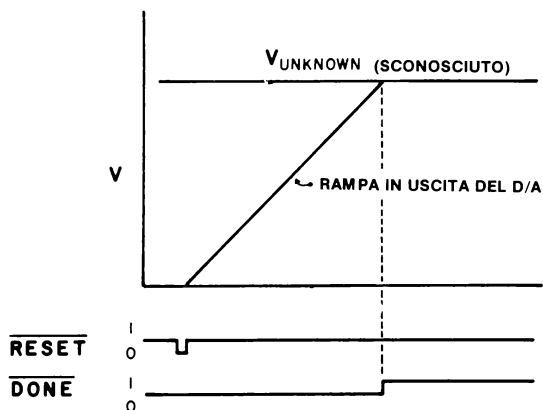



Figura 2-2. Diagramma dei tempi per un convertitore A/D analogo a quello riportato in Figura 2-1.

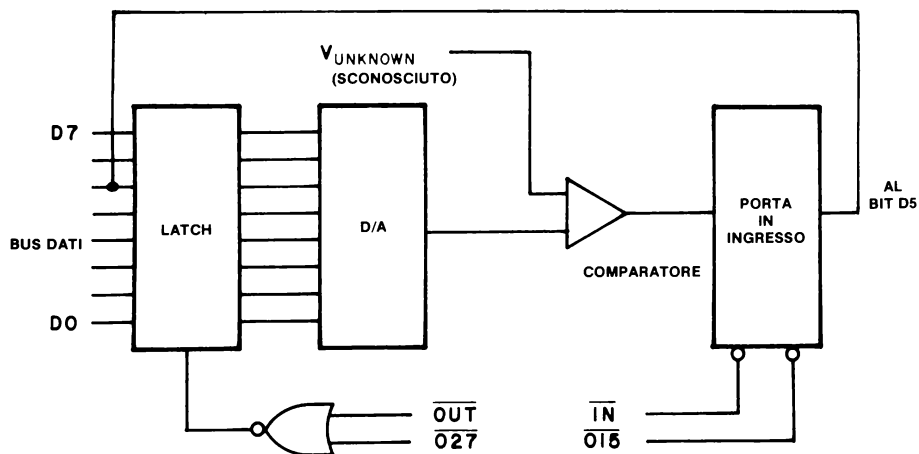



Figura 2-3. Tipico convertitore a rampa con controllo software che impiega un convertitore D/A, una porta di ingresso e un comparatore.

ESEMPIO 2-1
PROGRAMMA PER LA GENERAZIONE DI UNA RAMPA NEL CASO DI IMPIEGO DI UN CONVERTITORE E DI UN COMPARATORE

```
                        * 003 000
003 000 006   START,    MVIB        PONE IL REGISTRO B = 0
003 001 000             000
003 002 170   DALOOP,   MOVAB       SPOSTA IL VALORE DEL REGISTRO B IN A
003 003 323             OUT         E LO EMETTE SUL CONVERTITORE D/A
003 004 027             027
003 005 333             IN          LEGGE LO STATO DEL COMPARATORE
003 006 015             015
003 007 346             ANI         MASCHERA I BIT NON SIGNIFICATIVI
003 010 040             040
003 011 302             JNZ         ANCORA = 0?
003 012 020             DONE        NO, CONFRONTO VALIDO
003 013 003             0
003 014 004             INRB        SI, INCREMENTA DI 1 ED ESEGUE DI NUOVO
003 015 303             JMP
003 016 002             DALOOP
003 017 003             0
003 020 170   DONE,     MOVAB       TERMINATO, PRENDE IL VALORE
003 021 323             OUT         E LO VISUALIZZA IN USCITA
003 022 002             002
003 023 166             HLT         STATO DI HALT
```

Il programma mostrato nell'Esempio 2-1 viene impiegato con il circuito mostrato nella Figura 2-3, ma la sua esecuzione è lenta in quanto può impiegare fino a 7 millisecondi per compiere una conversione; infatti, può essere necessaria la ripetizione fino a 255 volte della parte di programma DALOOP, a seconda del valore di tensione da misurare. Un valore di tensione più alto richiederà un tempo di conversione più lungo che non un valore minore dal momento che la rampa dovrà essere incrementata più volte per raggiungere la condizione di uguaglianza.

Il tempo di conversione può assumere valori notevoli nel caso di convertitori a più alta risoluzione come ad esempio a 10, 12 e 14 bit. Inoltre il calcolatore viene completamente dedicato a questa funzione per i molti millisecondi necessari al programma per effettuare la conversione. Anche nel caso di variazioni lente del segnale, la conversione può non essere sufficientemente veloce in modo da "catturare" la porzione di segnale che si desidera misurare.

Per esempio, si può pensare di utilizzare un rilevatore di picco per segnalare che l'uscita di uno strumento abbia raggiunto un punto di massima in modo da poter essere digitalizzato. Il rilevatore di picco può essere realizzato in maniera sofisticata oppure semplicemente utilizzando un temporizzatore che indichi al convertitore o al microcalcolatore che deve digitalizzare il segnale in uscita allo strumento. Questo tipo di relazione è mostrato in Figura 2-4.

Il segnale $\overline{RESET}$ è generato dallo strumento per indicare sia al convertitore D/A a rampa che al calcolatore, che è stato raggiunto il massimo e che in quell'istante il valore di tensione che vogliamo digitalizzare è presente al punto *a*.
Il convertitore o il calcolatore acquisiscono l'informazione e inizia la generazione della rampa di tensione, ma dal momento che la rampa in uscita cambia lentamente troveremo che durante il tempo di generazione anche l'uscita dello strumento è cambiata.

La coincidenza delle due uscite, quella della rampa e quella dello strumento, si effettuerà al punto *b*, un certo tempo dopo il valore di picco. Se si prende il valore di questo istante

come valore di picco da digitalizzare, la misura che effettueremo sarà sbagliata di un valore notevole dal momento che la misura è stata effettuata quando il valore di picco non è più presente.

La tecnica di conversione A/D a rampa a controllo di programma è lenta e non è il metodo più usato per realizzare la conversione con i microcalcolatori.
La tecnica di conversione A/D a rampa rimane tuttavia valida ed esistono moduli di conversione standard che hanno tempi di conversione relativamente veloci ed un basso costo; queste due caratteristiche li rendono ideali per l'impiego per esperimenti in laboratorio e per uso generale.

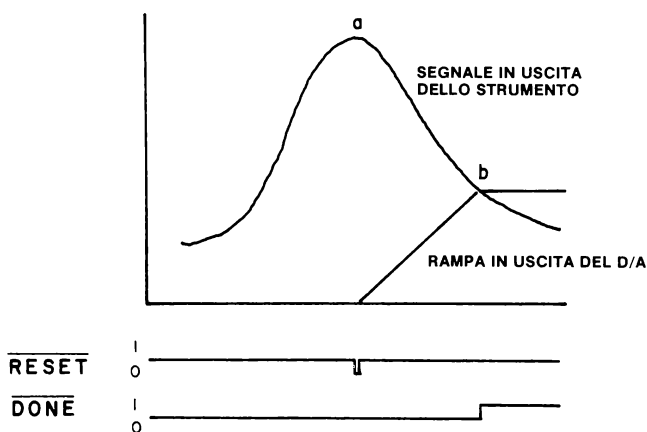



Figura 2-4. Diagramma dei tempi che mette in relazione l'uscita di uno strumento e il tentativo da parte di un convertitore D/A di digitalizzare il valore di picco.

Tre modelli sono:

Analog Devices, Inc., Norwood, MA 02062

ADC 8S	a 8 bit con tempo di conversione di 1 msec

Datel Systems, Inc., Canton, MA 02021

ADC Econoverter	a 6 bit con tempo di conversione di 50 μsec
ADC 98A	a 8 bit con un tempo di conversione di 20 μsec

Le tecniche di interfacciamento sia hardware che software richieste per il controllo dei moduli di conversione A/D sono le stesse di quelle discusse nel paragrafo relativo all'interfacciamento dei convertitori Analogico-Digitali di questo capitolo.

CONVERTITORI AD APPROSSIMAZIONI SUCCESSIVE

In molti casi, si vuole ottenere una possibilità di conversione che sia accurata e precisa, ma allo stesso tempo che comporti un tempo di esecuzioni di centinaia di microsecondi o addirittura dell'ordine di qualche millisecondo per conversioni di 8 e 10 bit. La

conversione ad approssimazioni successive permette di combinare in modo soddisfacente velocità e precisione.

Esamineremo qual'è il funzionamento di questa tecnica di conversione nel caso di individuazione di un valore sconosciuto nel campo da 0 a 255, possibile con parole da 8 bit. Se utilizzassimo la tecnica con generazione di rampa occorrerebbero molti passi per riconoscere il valore giusto, dal momento che ciascun passo realizza un incremento di una sola unità rispetto al valore precedente. Al contrario, il metodo ad approssimazioni successive è basato sul peso di ciascuna posizione binaria, partendo dal bit più significativo a quello meno significativo. Questo metodo vi costruisce internamente il valore sconosciuto con pochissime operazioni di test utilizzando un comparatore che indichi le condizioni di maggiore o minore.
I singoli test sono realizzati sommando o meno i pesi dei singoli bit ed il valore finale dipende dalla comparazione delle somme parziali con il numero sconosciuto. Di seguito presentiamo un esempio di conversione a 8 bit del numero 113 nell'equivalente binario.

VALORE SCONOSCIUTO = 113

Valore Test	*Esito della comparazione*	*Somma*
128	Troppo alto, da non aggiungere nella somma	0
64	Troppo basso, aggiungere nella somma e continuare	64
32	64+32 è ancora troppo basso, aggiungere 32 e continuare	96
16	64+32+16 è ancora troppo basso, aggiungere 16 e continuare	112
8	La somma è troppo alta, non aggiungere questo valore	112
4	La somma è troppo alta, non aggiungere questo valore	112
2	La somma è troppo alta, non aggiungere questo valore	112
1	La somma 64+32+16+1 è il valore ricercato	113

Dal momento che i pesi di ciascun bit sono stati utilizzati nell'approssimazione, diventa semplice convertire l'informazione della tabella in una rappresentazione binaria del numero 113 decimale ponendo il valore 1 nelle posizioni che hanno contribuito alla somma, e 0 a quelle che ne sono state escluse.

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
128	64	32	16	8	4	2	1
0	1	1	1	0	0	0	1

La tecnica ad approssimazioni successive comporta l'esecuzione di soli 8 test per trovare il valore da misurare contro i 113 del metodo a rampa; generalizzando, il metodo richiede un test per bit, per cui nel caso di conversione di valori a 10 bit, si hanno 10 test per un campo di variazione di 1023 valori diversi.

È necessario comunque avere a disposizione un convertitore per generare i valori di tensione corrispondenti ai pesi dei singoli bit da utilizzare per i test; il controllo di questo convertitore può essere realizzato sia hardware che software.
Affronteremo il caso di impiego di una soluzione software mentre trascureremo quella hardware.

L'interfaccia con il calcolatore usata nell'esempio è la stessa di quelle utilizzate nel caso di conversione a rampa di controllo di programma (Figura 2-3).

Non è necessaria nessuna modifica di tipo circuitale dal momento che rimangono le stesse sia la tecnica di emissione dei valori digitali sia quella di acquisizione del valore dell'uscita del comparatore.

Il metodo o *algoritmo* di programma è di facile interpretazione. Il calcolatore inizia ponendo tutti i bit all'ingresso del convertitore al livello logico zero, dopo di che pone il bit più significativo (MSB) a 1 ed esegue il test sull'uscita del comparatore. Se il convertitore ha emesso un valore di tensione superiore a quella da misurare, il comparatore presenterà alla sua uscita il valore logico 1 e di conseguenza il calcolatore riporterà il valore logico del bit considerato a zero. Analogamente verrà mantenuto a 1 la posizione considerata se l'uscita del comparatore è zero, indicando cioé che il valore emesso dal convertitore è inferiore alla tensione sconosciuta.
A questo punto si esegue la stessa operazione con il bit meno significativo successivo fino ad esaurimento delle posizioni.

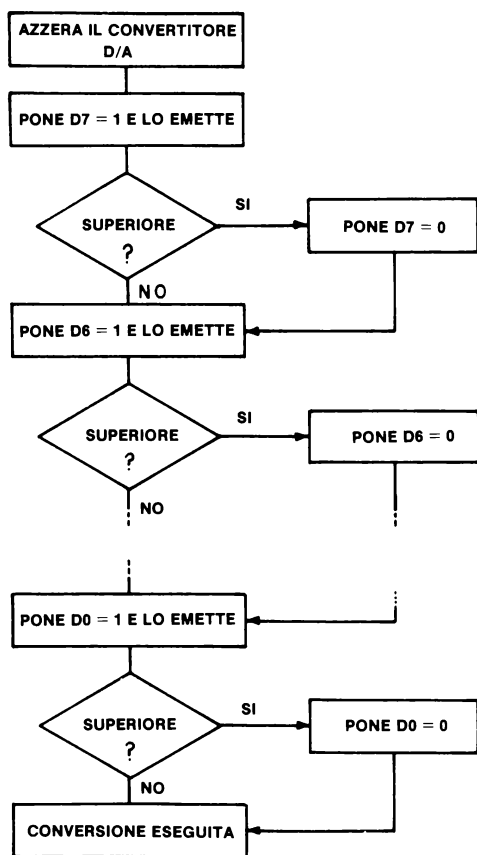



Figura 2-5. Schema a blocchi di un programma per la conversione di valori a 8 bit con il metodo delle approssimazioni successive.

Lo schema a blocchi di Figura 2-5 mostra come il programma potrebbe operare. Mentre a prima vista può sembrare un lungo e complicato programma, l'impiego di istruzioni logiche, di rotazione e cicli di programma lo rendono di semplice scrittura.
È necessario tuttavia l'impiego di parecchi registri dell'8080 con le seguenti funzioni:

- Registro A — per il trasferimento di I/O e manipolazione dei bit
- Registro B — per la memorizzazione del valore di test
- Registro C — per la memorizzazione del risultato che si forma man mano
- Registro D — per il conteggio dei cicli.

Potete seguire lo sviluppo del programma e il contenuto assunto dai vari registri con carta e matita, immaginando di confrontare un valore qualsiasi fissato da voi. Oppure, potete utilizzare una telescrivente o un terminale se disponete del programma *"DBUG: An 8080 Interpretive Debugger"*, contenuto nella serie di Applicazioni dei Bugbook. Qui di seguito mostriamo il programma completamente assemblato:

```
                        ESEMPIO 2-2
                        PROGRAMMA PER LA CONVERSIONE A/D AD
                        APPROSSIMAZIONE SUCCESSIVA PER VALORI A 8 BIT

                        *003 000
003 000 227   START,    SUBA      AZZERA A
003 001 001             LXIB      CARICA LA COPPIA DI REGISTRI B,C
003 002 000             000       CON C = 0
003 003 200             200       E B = 10000000
003 004 026             MVID      INIZIALIZZA IL CONTATORE CON IL VALORE 8
003 005 010             010
003 006 260   AGAIN,    ORAB      ESEGUE L'OR TRA A E B
003 007 117             MOVCA     MEMORIZZA IL RISULTATO IN C
003 010 323             OUT       E LO EMETTE DUL DAC
003 011 027             027
003 012 333             IN        LEGGE LO STATO DEL COMPARATORE
003 013 001             001
003 014 346             ANI       MASCHERA I BIT NON NECESSARI
003 015 040             040
003 016 312             JZ        E' MAGGIORE DEL VALORE SCONOSCIUTO?
003 017 025             OK        NO, VA A OK
003 020 003             0
003 021 170             MOVAB     SI,PRENDE IL VALORE USATO PER IL TEST
003 022 057             CMA       LO COMPLEMENTA
003 023 241             ANAC      PONE IL BIT TESTATO A ZERO
003 024 117             MOVCA     MEMORIZZA IL DATO IN C
003 025 170   OK,       MOVAB     PRENDE IL VALORE USATO PER IL TEST
003 026 037             RAR       LO RUOTA A DESTRA DI UNA POSIZIONE
003 027 107             MOVBA     E LO PONE NUOVAMENTE IN B
003 030 171             MOVAC     PONE IL DATO SUL REGISTRO A
003 031 025             DCRD      DECREMENTA IL CONTATORE DI CICLI, D= 0?
003 032 302             JNZ       NO, ESEGUE IL TEST CON IL PROSSIMO BIT
003 033 006             AGAIN
003 034 003             0
003 035 166   DONE,     HLT       TERMINA CON VALORE FINALE IN A.
```

Questo programma esegue una conversione ad approssimazioni successive a 8 bit; per numero di bit superiori il programma si complica dal momento che è necessario l'impiego di parole di più di un byte sia per i test che per il risultato. Una conversione ad

approssimazioni successive a 10 bit è riportata nel Capitolo 7; impiegando evidentemente un convertitore D/A a 10 bit.

Utilizzando il programma dell'Esempio 2-2, una conversione a 8 bit richiede un tempo di esecuzione di circa 240 microsecondi (con clock dell'8080 di 500 nanosecondi) qualunque sia il valore da convertire, dal momento che tutte le posizioni vengono testate. Il campionamento e la digitalizzazione possono essere rappresentati nel modo illustrato nella Figura 2-6

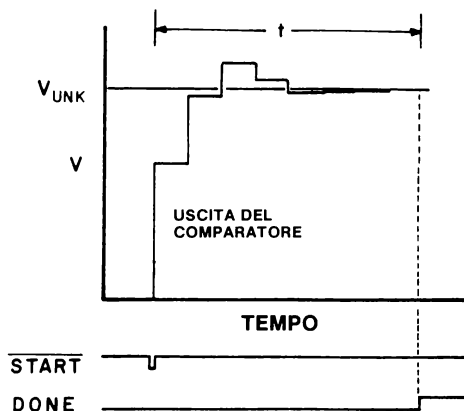



Figura 2-6. Diagramma dei tempi per la conversione A/D ad approssimazioni successive.

Se osservate attentamente la Figura 2-6, noterete che alcuni valori emessi dal convertitore D/A si pongono al di sopra, altri al di sotto del valore da misurare e che il meccanismo di confronto termina solo quando il valore "costruito" all'interno del programma si sovrappone a quello sconosciuto.

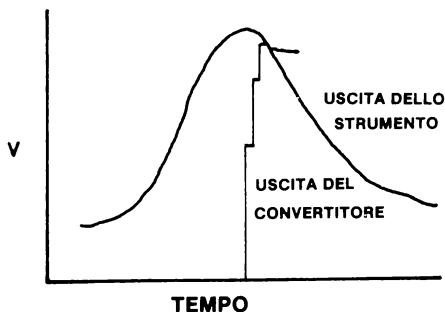



Figura 2-7. Relazione temporale tra l'uscita di uno strumento e il risultato di un programma di conversione ad approssimazioni successive per digitalizzare il picco di tensione.

La tecnica di conversione, appena mostrata può essere utilizzata nel caso dell'esempio contenuto nella Figura 2-4, dove si voleva digitalizzare un picco di tensione generato da uno strumento con una tecnica di conversione a rampa. Nel caso di conversione più veloce come quella ad approssimazioni successive, la soluzione del problema è quella mostrata nella Figura 2-7

Tuttavia, la conversione anche adottando questa nuova tecnica è troppo lenta per catturare il valore di picco nel suo valore massimo, di modo che il valore ottenuto è inferiore a quello reale. Esiste una differenza importante tra il funzionamento in un caso e nell'altro. Nel caso della rampa la conversione termina quando il valore di rampa e quello in uscita dello strumento coincidono. Nel caso delle approssimazioni successive si verifica una coincidenza di valori, ma la conversione sta ancora eseguendo dei test sugli ultimi bit. *Il processo di conversione continua fino a quando viene effettuato il test su tutti i bit.* Ciò significa che il test sugli ultimi 4 o 5 bit viene effettuato confrontando un valore decrescente dell'uscita dello strumento, in modo da ottenere una notevole differenza tra il valore finale prodotto dall'algoritmo del programma e quello presente, al termine della conversione, sull'uscita dello strumento. Questo è un problema reale e deve essere tenuto in considerazione quando esiste la possibilità che il segnale cambi più velocemente dalla capacità del convertitore di eseguire il confronto. Ci sono vari modi per ovviare a questo inconveniente come vedremo in seguito.

Ci sono alcune applicazioni nelle quali può essere effettivamente utile l'impiego di convertitori A/D ad approssimazioni successive a controllo di programma; tuttavia esistono moduli di conversione che adottano questa tecnica ed hanno velocità di conversione notevole ed un basso costo. Ne riportiamo alcuni di tre produttori diversi:

Analog Devices, Inc., Norwood, MA 02062

ADC 10Z	10 bit	tempo di conversione 10 μs
AD 7570	10 bit	tempo di conversione 20 μs

Burr-Brown Research Corp Tucson, AZ 85734

ADC80AG-10	10 bit	tempo di conversione 18 μs
ADC80AG-12	12 bit	tempo di conversione 25 μs

Datel Systems, Inc., Canton MA 02021

ADCHY12BC	12 bit	tempo di conversione 8 μs

Con l'eccezione del convertitore AD7570 dell'Analog Devices, tutti gli altri hanno la possibilità di diverse tensioni in ingresso. I valori più comuni sono± 2,5 volt, ± 5 volt± 10 volt, 0-5 volt; 0-10 volt. L'AD7570 è particolarmente interessante dal momento che un circuito monolitico realizzato con tecnologia CMOS ed ha uscita di tipo three-state; inoltre è facilmente interfacciabile con sistemi a microprocessore. Nonostante molti moduli dispongono di codici in uscita diversi, nel nostro libro considereremo solo quelli che adottano codici binari o BCD.

È stata nostra intenzione in questi paragrafi di mostrare come funzionano i metodi di conversione a rampa e ad approssimazioni successive e quali siano i programmi necessari per il loro impiego. Molti probabilmente saranno più interessati all'interfacciamento di qualcuno di questi moduli piuttosto che a quelli realizzati con controllo di programma. Nei prossimi paragrafi di questo capitolo descriveremo i circuiti di interfaccia e i programmi usati per l'interfacciamento modulare con convertitori A/D.

INTERFACCIAMENTO DI MODULI DI CONVERSIONE A/D

Abbiamo già accennato al fatto che molti possono preferire l'impiego di moduli di conversione A/D già pronti piuttosto che ricorrere a soluzioni come quelle della conversione a controllo di programma, comunque particolarmente adatte in alcune applicazioni. Affronteremo quindi l'interfacciamento con due convertitori A/D uno a 8 bit e l'altro a 10; mostreremo cosa si può fare con essi ed altre cose come la visualizzazione di dati organizzati in file e le temporizzazioni dell'acquisizione di dati.

INTERFACCIAMENTO DI UN CONVERTITORE A 8 BIT

La maggior parte dei convertitori A/D non esegue conversioni una dopo l'altra alla massima velocità loro consentita, per cui occorre più tempo per l'acquisizione di un valore dal mondo esterno che per la sua richiesta e il suo trasferimento al calcolatore. In generale i convertitori eseguono una conversione singola quando il calcolatore richiede quest'operazione mediante un segnale che è costituito da un impulso di START o di CONVERT (conversione in corso) compatibili TTL, che viene facilmente generato sotto controllo software, e può essere sia uno zero logico che un uno logico, a seconda dello specifico modulo di conversione scelto.

Poiché il convertitore A/D richiede un certo tempo per effettuare la conversione che varia da pochi microsecondi a molti millisecondi, non si può pensare che un segnale sconosciuto sia digitalizzato non appena l'implulso di START viene ricevuto dal

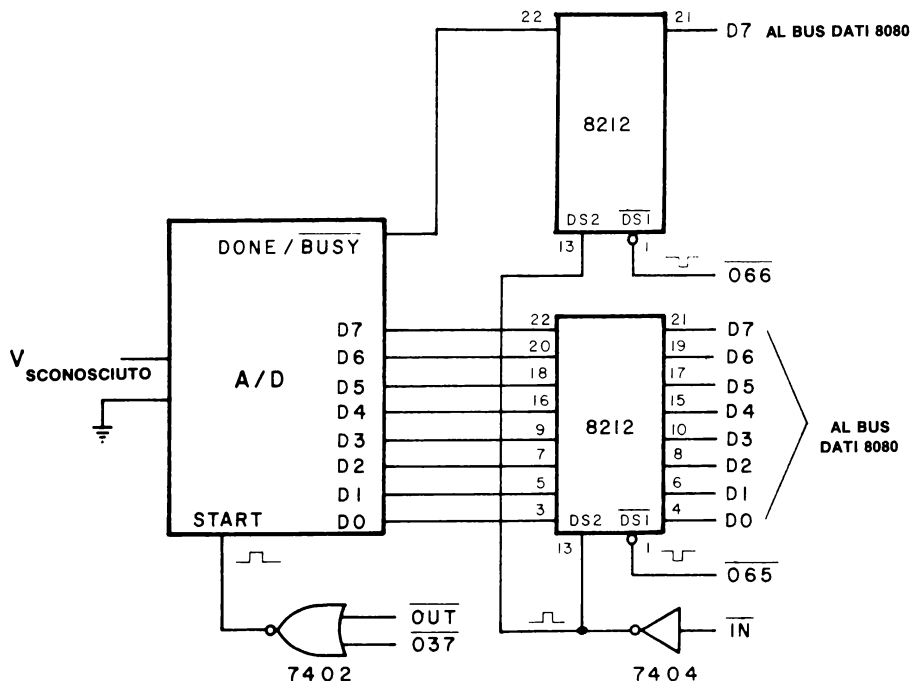



Figura 2-8. Un tipico convertitore A/D a otto bit interfacciato con un microcalcolatore. I collegamenti avvengono con il bus dati.

convertitore A/D. Nella maggior parte dei casi il convertitore A/D avrà un segnale di *stato* o flag dell'uscita, che indicano che la conversione è in atto (stato BUSY) oppure che è stata eseguita (DONE). In questo secondo caso, le uscite digitali sono l'esatta rappresentazione della tensione sconosciuta.

L'interfaccia tipica di un convertitore A/D a otto bit richiede una porta di ingresso three-state a otto bit per i dati, una porta di ingresoo three-state per il flag DONE/$\overline{BUSY}$ e un impulso di uscita per far partire il convertitore. Nella figura 2-8 è illustrato un esempio di interfaccia tipica. Il software che può essere usato per controllare il modulo è indicato invece nell'Esempio 2-3 in cui il programma è scritto in forma di sottoprogramma che fa partire la conversione all'interno del modulo emettendo un impulso sull'ingresso di START e che controlla lo stato del flag DONE/$\overline{BUSY}$. Non appena la conversione è avvenuta, il valore a otto bit viene immesso nel registro A dell'8080.

Il sottoprogramma non memorizza i dati nella memoria del calcolatore ma lo lascia semplicemente nel registro A. Per far sì che si abbia anche la memorizzazione, se necessaria, occorrerà aggiungere nuovi passi di programma. Di fatto, le interfacce e il software per i convertitori A/D a otto bit sono piuttosto semplici.

I convertitori A/D a otto bit hanno un campo di applicazione ben definito, e gli utenti che necessitano di una maggiore *risoluzione* o hanno bisogno di dividere i loro segnali in porzioni più piccole, devono ricorrere ai convertitori A/D a 10 o 12 bit, più complessi da interfacciare dato che per il trasferimento e la memorizzazione dei dati sono necessari byte multipli. Benché questi dispositivi a più byte rappresentino il caso più comune, abbiamo scelto di andare a fondo in modo più dettagliato in questi ultimi che in quelli a otto bit.

```
                              ESEMPIO 2-3
                              TIPICO SOTTOPROGRAMMA DEL CONTROLLORE ADC A
                              8 BIT
                              VALORE BINARIO A 8 BIT RIPRISTINATO NEL
                              REGISTRO A

                              * 000 000
000 000 323     CONV,         OUT         FA PARTIRE IL CONVERTITORE
000 001 037                   037
000 002 333     TEST,         IN          LEGGE IL BIT DI STATO
000 003 066                   066
000 004 346                   AN!         MASCHERA TUTTI I BIT NON DESIDERATI
000 005 200                   200
000 006 312                   JZ          È FATTO? ESEGUITO?
000 007 002                   TEST        NO, FLAG= 0, RIPETI IL TEST
000 010 000                   0
000 011 333                   IN          SI, FLAG= 1, CARICA I DATI
000 012 065                   065
000 013 311                   RET         ESEGUITO, RITORNA CON IL VALORE IN A
```

INTERFACCIAMENTO DI UN CONVERTITORE A/D A DIECI BIT

I convertitori A/D a 10 bit vengono usati in situazioni in cui è necessaria una risoluzione su base 1024. L'interfacciamento di questo tipo di convertitori con un microcalcolatore non è difficile in quanto vengono usati gli stessi elementi di interfacciamento usati per i convertitori a otto bit: per la immissione di dati e per le informazioni su flag vengono usate porte three-state e anche in questo caso viene usato un impulso di uscita per attivare il processo di conversione.

L'input e l'output di parole di dati più lunghe di otto bit richiedono il trasferimento di dati in due o più byte a otto bit. Questo caso è stato ampiamente illustrato nella Parte 1 in un esempio su un convertitore a 10 bit, in cui erano stati usati due byte a otto bit di memoria per trasferire i dati in un convertitore A/D a dieci bit, nonostante fossero stati usati solo due bit in una parola di dati.

Quando si intende interfacciare un convertitore A/D a dieci bit ad un calcolatore, si può scoprire che è necessaria una porta di ingresso addizionale per i due bit di dati in più. In questo caso possiamo usare lo stesso tipo di interfaccia usato con il convertitore A/D a otto bit dato che esistono bit non utilizzati e disponibili su una delle porte di ingresso. L'interfaccia completa è illustrata nella Figura 2-9.

Poiché devono essere caricati due byte di dati, occorre adattare il software usato per controllare il convertitore A/D. A questo scopo l'Esempio 2-4 fornisce un completo listing.

L'uscita del flag di stato del convertitore può essere collaudata nello stesso modo seguito nel caso del convertitore a otto bit, con un'istruzione AND usata in quel caso per mascherare i bit non desiderati. Comunque se due dei bit inutilizzati della porta di input three-state usata per l'input della condizione del flag vengono usati per caricare i dati, l'operazione AND "distrugge" i due bit di dati e questa operazione dovrà essere caricata un'altra volta. Per sostituire l'istruzione AND è stata scelta un'istruzione di addizione. Aggiungendo un uno al bit più significativo, si avrà un aumento di capacità o un carry quando l'MSB è un uno, mentre se K l'MSB è uno zero, non si avrà *carry*. È questa

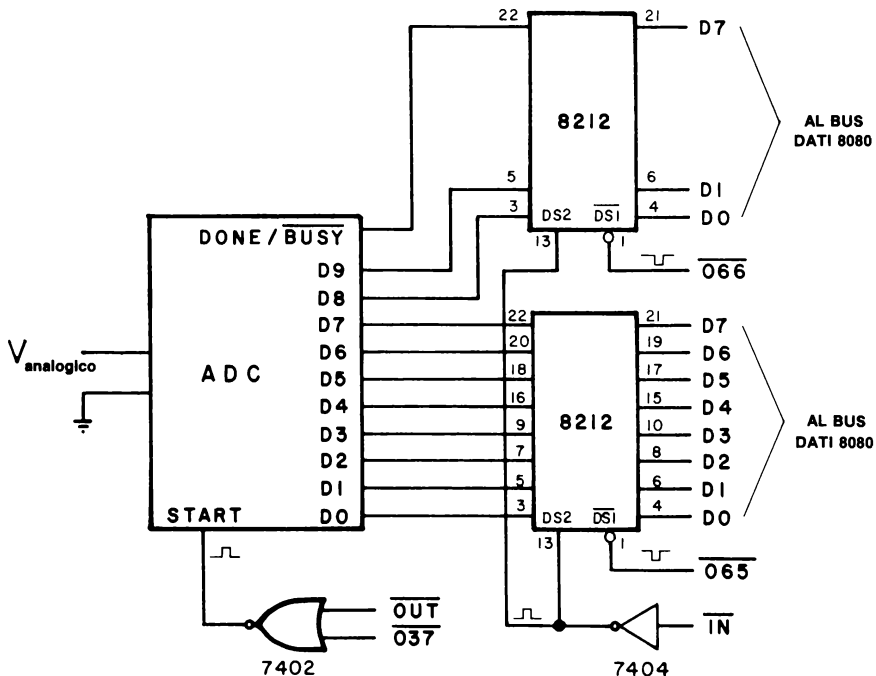



Figura 2-9. Un tipico convertitore A/D a dieci bit interfacciato con un microcomputer a otto bit mediante bus di dati.

un'operazione particolarmente utile dato che gli zeri vengono aggiunti alla altre posizioni di bit. Va notato che 0 + X =X, per cui gli altri sette bit non saranno intaccati da quest'operazione di addizione.

Le istruzioni in più sono state incorporate nel software illustrato nell'Esempio 2-4 dove i dati del convertitore A/D vengono memorizzati nei registri B e C, che contengono rispettivamente i due bit più significativi del convertitore A/D nelle posizioni D1 e D0 e gli altri otto bit nelle posizioni da D7 a D0.

Il programma contiene inoltre le istruzioni PUSH e POP in modo da conservare il contenuto del registro A e i flag durante il processo di controllo del convertitore A/D. In casi come questi, è utile far chiamare il sottoprogramma ADC dal programma principale e poi memorizzare i dati, piuttosto che far eseguire tutte queste funzioni dal sottoprogramma ADC, e ciò per mantenere quest'ultimo il più generico possibile in modo che anche le altre parti del programma possano accedervi.

ESEMPIO 2-4
ROUTINE DI INGRESSO ADC PER UN CONVERTITORE A 10 BIT.
I DUE MSB SONO CARICATI NEL REGISTRO B E GLI LSB NEL REGISTRO C

```
                         * 003 000
003 000 365    ADC,      PUSHPSW    MEMORIZZA REGISTRO A E I FLAG
003 001 323              OUT        INIZIA UNA CONVERSIONE
003 002 037              037
003 003 333    TEST,     IN         LEGGE LO STATO & 2 MSB
003 004 066              066
003 005 306              ADI        AGGIUNGE 1 AL BIT DEL FLAG PER
003 006 200              200        EFFETTUARE UN RIPORTO NESSUN RIPORTO,
003 007 322              JNC        CONTROLLA DI NUOVO
003 010 003              TEST
003 011 003              0
003 012 107              MOVBA      FLAG=1, ALLORA MEMORIZZA GLI MSB
003 013 333              IN         CARICA GLI LSB
003 014 065              065
003 015 117              MOVCA      LI MEMORIZZA NEL REGISTRO C
003 016 361              POPPSW     RIMEMORIZZA IL REGISTRO A E I FLAG
003 017 311              RET        RITORNA
```

Con gli Esempi 2-5 e 2-6 ecco altri due programmi, in cui vengono fornite delle istruzioni per memorizzare nella memoria a lettura/scrittura i dati ricevuti dal convertitore A/D a 10 bit. Il programma nell'Esempio 2-5 illustra uno dei modi in cui ciò può essere effettuato. Se i registri B e C contengono i valori richiesti prima di chiamare il sottoprogramma i dati immagazzinati in essi sarebbero perduti dato che i registri vengono usati anche nel sottoprogramma ADC per una memorizzazione temporanea di dati.

ESEMPIO 2-5
QUESTO PROGRAMMA CHIAMA IL SOTTOPROGRAMMA "ADC" QUINDI MEMORIZZA I 10 BIT DI DATI IN UN FILE CONTENUTO NELLA MEMORIA A LETTURA/SCRITTURA

```
                         * 004 000
004 000 061              LXISP      CARICA UN VALORE DEL PUNTATORE DI STACK
004 001 000              000
004 002 060              060
```

```
004 003 041    LXIH     CARICA IL PUNTATORE DI MEMORIA
004 004 000    000      INDIRIZZO PARTE BASSA = 000
004 005 061    061      INDIRIZZO PARTE ALTA = 061
004 006 315    CALL     CHIAMA LA ROUTINE DI CONVERSIONE
004 007 000    ADC
004 010 003    0
004 011 161    MOVMC    MEMORIZZA GLI 8 BIT LSB IN MEMORIA
004 012 043    INXH     INCREMENTA L'INDIRIZZO
004 013 160    MOVMB    MEMORIZZA GLI 8 MSB
004 014 043    INXH     INCREMENTA DI NUOVO
004 015 000    0        IL SOFTWARE IN PIU' CONTINUERA'
                        DA QUESTO PUNTO
```

Il programma illustrato nell'Esempio 2-6 illustra il modo in cui le istruzioni PUSH e POP possono essere usate nel *programma principale* per memorizzare il contenuto dei registri B e C e quindi per rimemorizzarne i valori dopo che il sottoprogramma ADC e la parte di memorizzazione dati del programma principale hanno completato la loro esecuzione. Le istruzioni PUSHB e POPB non possono essere usate nel sottoprogramma ADC se vogliamo avere accesso al valore a 10 bit proveniente dal convertitore A/D. Nel caso in cui lo fossero si avrebbe la seguente sequenza qualora si ricorresse al sottoprogramma ADC; i valori già presenti nei registri B e C verrebbbero allora memorizzati nello stack, i registri B e C verrebbero usati per memorizzare i dati provenienti dal convertitore A/D, l'operazione di "pop" ripristinerebbe i vecchi valori nei registri B e C e l'operazione di ritorno restituirebbe il controllo al programma principale. In questo modo i dati provenienti dal convertitore A/D non sarebbero disponibili. È sempre molto importante prestare la massima attenzione quando si eseguono operazioni con lo stack. Ricordate che:*il puntatore di stack deve essere sistemato in un'area disponibile della memoria a lettura/scrittura prima di ricorrere a qualsiasi operazione con lo stack.*

ESEMPIO 2-6
QUESTO SOFTWARE ESEGUE UNA CONVERSIONE E LA MEMORIZZAZIONE DI DATI, MANTENDENDO INALTERATI I REGISTRI B E C

```
               * 004 000
004 000 061    LXISP    CARICA IL PUNTATORE DI STACK
004 001 000    000
004 002 060    060
004 003 041    LXIH     CARICA IL PUNTATORE ALLA MEMORIA
004 004 000    000
004 005 061    061
004 006 305    PUSHB    MEMORIZZA I REGISTRI B E C SULLO STACK
004 007 315    CALL     CHIAMA LA ROUTINE DI CONVERSIONE
004 010 000    ADC
004 011 003    0
004 012 161    MOVMC    QUESTA MEMORIZZA I DATI
004 013 043    INXH
004 014 160    MOVMB
004 015 043    INXH
004 016 301    POPB     QUESTA RIPRISTINA I REGISTRI B E C
004 017 000    0        QUI DOVRA' ESSERE AGGIUNTO DEL
                        SOFTWARE
```

L'USO DI CONVERTITORI A/D PER L'ACQUISIZIONE DI DATI

L'uso di convertitori analogico-digitali con i microcalcolatori permette l'acquisizione di dati ad una velocità massima di circa 20.000 punti al secondo. Questa approssimazione è stata fatta usando un convertitore a otto bit con un semplice programma di input e passi di programma che dovrebbero essere usati per memorizzare i dati ed eseguire varie operazioni di software di supervisione "overhead" come il conteggio del numero di punti di dato acquisiti. I dati non possono essere acquisiti per un tempo lungo se non si può variare questa velocità di acquisizione poiché il computer andrebbe fuori dalla memoria di lettura/scrittura nella quale i dati vanno memorizzati. Per molte misure è più ragionevole puntare la velocità a valori compresi tra 10 e 25 punti al secondo.

In questo paragrafo esamineremo l'uso di un piccolo calcolatore per acquisire dati alla temperatura di una serra per un periodo di 24 ore. La temperatura viene rilevata ogni 10 *minuti* usando un convertitore A/D a 10 bit interfacciato secondo lo schema già illustrato nella Figura 2-9.

Supponiamo che il sensore sia normalizzato per avere un'uscita di 0-5 volt, cioé il campo dell'ingresso analogico di un convertitore A/D. In questo caso possiamo usare una termocoppia o un circuito di ponte, oppure, molto più semplicemente dei dispositivi allo stato solido come il chip di controllo della temperatura LM3911 della National Semiconductor, che fornisce un'uscita di tensione direttamente proporzionale ai gradi centigradi (Celcius).

Poiché i dati provenienti dal sensore della temperatura devono essere acquisiti per un periodo di 24 ore con un intervallo di 10 minuti tra ogni due punti, il convertitore A/D a 10 bit, dovrà effettuare 144 misure. Ogni valore binario a 10 bit richiederà due locazioni di memoria a otto bit per la memorizzazione. Il computer deve memorizzare i dati per il periodo di 24 ore e poi deve fermarsi per dar modo ai dati di essere esaminati.

A questo scopo abbiamo scritto un programma di ritardo di 10 minuti che verrà usato per "ritardare" il calcolatore tra due valori di temperatura. Il tempo effettivo necessario per eseguire i passi di acquisizione di dati e i passi di memorizzazione più il tempo necessario al convertitore per eseguire la conversione è molto limitato (da 100 a 200 microsecondi) se paragonato ai 10 minuti che intercorrono tra due letture di temperatura. Il programma di ritardo di 10 minuti è un programma generale e non tiene conto di questi periodi così brevi. Nell'Esempio 2-7 è illustrato un programma di acquisizione dati completamente assemblato.

ESEMPIO 2-7
PROGRAMMA DI ACQUISIZIONE DATI PER UN ADC A 10 BIT
CHE CAMPIONA 144 PUNTI A UN PUNTO OGNI 10 MINUTI

```
                           * 003 000
003 000 061   START,      LXISP       CARICA IL PUNTATORE DI STACK PER
                                      SOTTOPROGRAMMI
003 001 377               377
003 002 003               003
003 003 026               MVID        CARICA D COME CONTATORE A 144 PUNTE
003 004 220               220
003 005 041               LXIH
003 006 000               000
003 007 002               002
003 010 323   CONVRT,     OUT         FA PARTIRE UNA CONVERSIONE
003 011 037               037
003 012 333   TEST,       IN          LEGGE IL BIT DI FLAG
003 013 066               066
003 014 306               ADI         AGGIUNGE 1 AL BIT DEL FLAG PER
003 015 200               200         EFFETTUARE UN RIPORTO NESSUN RIPORTO.
```

```
003 016 322            JNC        LO CONTROLLA DI NUOVO
003 017 012            TEST
003 020 003            0
003 021 107            MOVBA      MEMORIZZA GLI MSB
003 022 333            IN         CARICA OTTO LSB
003 023 065            065
003 024 167            MOVMA      LI CARICA IN MEMORIA
003 025 043            INXH       INCREMENTA IL PUNTATORE ALLA MEMORIA
003 026 160            MOVMB      MEMORIZZA ANCHE I DUE MSB
003 027 043            INXH
003 030 025            DCRD       DECREMENTA IL CONTATORE DI PUNTI
003 031 312            JZ         È UGUALE A 0?
003 032 073            DONE       SE SI', ESEGUE
003 033 003            0
003 034 315            CALL       NO, CHIAMA IL RITARDO DEL TIMER
003 035 042            DELAY
003 036 003            0
003 037 303            JMP        TORNA INDIETRO E LO ESEGUE DI NUOVO
003 040 010            CONVRT
003 041 003            0
```

```
                                  QUESTO E' UN SOTTOPROGRAMMA DI
                                  RITARDO DA 10 MINUTI
003 042 365   DELAY,   PUSHPSW    SALVA I REGISTRI E I FLAG
003 043 305            PUSHB
003 044 325            PUSHD
003 045 021            LXID       ESEGUI 1200 CICLI DI 0,5 SECONDI PER
003 046 260            260        CIASCUN CICLO DI RITARDO
003 047 004            004
003 050 001   TIME1,   LXIB       CICLO DI RITARDO DI 0,5 SEC.
003 051 303            303        COSTANTI PER CLOCK DI 500 ns
003 052 242            242
003 053 013   TIME2,   DCXB       DECREMENTA LA COPPIA DI REGISTRI B E C
003 054 170
003 055 261            ORAC       ESEGUE L'OR DI B CON C
003 056 302            JNZ        SE NON È=0 RIPETE
003 057 053            TIME2
003 060 003            0
003 061 033            DCXD       DECREMENTA IL SECONDO CONTATORE
003 062 172            MOVAD      PRENDE D
003 063 263            ORAE       ESEGUE L'OR DI B CON C
003 064 302            JNZ        SE NON È FINITO TORNA INDIETRO A
003 065 050            TIME 1
003 066 003            0
003 067 321            POPD       SALVARE I REGISTRI E I FLAG
003 070 301            POPB
003 071 361            POPPSW
003 072 311            RET        RITORNA DOPO 600 SECONDI
                                  IL PROGRAMMA SI FERMA QUI QUANDO È
                                  ESEGUITO.
003 073 166   DONE,    HLT        FINE DEL PROGRAMMA.
```

TEMPORIZZAZIONE DELL'ACQUISIZIONE DATI

Il programma illustrato nell'Esempio 2-7 lavorerà acquisendo un valore di temperatura a 10 bit ogni 10 minuti, mentre il sottoprogramma DELAY impegnerà il calcolatore impedendogli di eseguire altre operazioni. In un programma come questo, il computer è *dedicato* al software di misura della temperatura, tuttavia esistono metodi hardware e software che aiutano ad eliminare questo inconveniente.

Scoprirete presto che i cicli di temporizzazione del software monopolizzano il tempo del calcolatore. Evidentemente il calcolatore non può eseguire altre operazioni software durante l'esecuzione di un programma di ritardo. Per ritardi brevi è possibile usare il calcolatore solamente per il ritardo del programma. Persino i ritardi lunghi come il sottoprogramma DELAY da 10 minuti, possono essere accettati se il calcolatore non ha altre operazioni da eseguire. Se il sistema di elaborazione utilizza degli interrupt e il programmatore si affida ai cicli di ritardo software, i ritardi effettivi potranno essere più lunghi di quelli previsti. Ciò succede quando i cicli di ritardo vengono interrotti da un dispositivo esterno. Poiché il software del dispositivo di interruzione richiede una durata ben definita di tempo di esecuzione del calcolatore, il ritardo complessivo diventa più lungo del previsto. Per alcuni sistemi questo è un punto molto importante.

Nell'esempio relativo alle misure della temperatura di una serra, non ci sono al momento dispositivi di interruzione collegati con il calcolatore. Per evitare che si presentino in futuro degli inconvenienti, sarebbe meglio considerare una alternativa al software di ritardo simile a quella usata nel sottoprogramma DELAY.

Invece del ciclo di ritardo software, per temporizzare gli intervalli da 10 minuti si userà un clock esterno in *tempo reale.* Questo tipo di clock è indipendente dal tempo di esecuzione del software e temporizzerà accuratamente i periodi che non saranno influenzati. Di solito si usano un oscillatore al quarzo e una serie appropriata di contatori e divisori digitali per fornire un'accurata serie di frequenze o periodi. Nella Figura 2-10 è illustrato un tipico clock.

I divisori o contatori addizionali (entrambi eseguono la stessa funzione), possono essere aggiunti al clock al quarzo per fornire altri periodi o frequenze. Il sistema di acquisizione di dati relativi alla temperatura della serra richiederebbe un periodo da 10 minuti, o 600 secondi, in modo che risultino necessari nuovi contatori. Molti sistemi veloci di acquisizione di dati richiedono solo le frequenze più alte.

I periodi di tempo forniti da un clock in tempo reale vengono "acquisiti" in uno dei due modi seguenti: o con un flag che viene testato a controllo di programma o con un interrupt indipendente dal controllo del programma. Entrambi i metodi sono importanti e verranno trattati perciò più ampiamente in seguito.

Figura 2-10. Schema a blocchi di un oscillatore al quarzo stabile e divisori di frequenza usati in un circuito di clock in tempo reale.

TEMPORIZZATORI TESTATI A CONTROLLO DI PROGRAMMA

In questo esempio di temporizzatore di flag a controllo di programma, un semplice flip-flop viene utilizzato per sentire il fronte positivo di una delle uscite del divisore del flop al quarzo. Il periodo di clock si può assumere sia 10 minuti, cosicchè si verifica un fronte positivo una sola volta ogni dieci minuti. I circuiti necessari per implementare un semplice temporizzatore testato a controllo di programma è mostrato in Figura 2-11.

Figura 2-11. Interfacciamento tipico per un temporizzatore testato a controllo di programma.Lo stato del flag è un impresso del calcolatore sul bus dei dati, il bit DO.

Un breve programma di esempio che illustra l'impiego di un circuito temporizzatore è riportato nell'Esempio 2-8. Quando questa parte del programma viene eseguita, lo stato del flag è letto e testato. Se l'uscita del flip-flop è ancora zero (conteggio in corso) il calcolatore esegue delle istruzioni che provocano il salto dei passi di programma che eseguono il controllo del convertitore fino all'etichetta CONT, dove prosegue l'esecuzione del programma. Se l'uscita del flip-flop è al valore logico 1 (tempo scaduto) il calcolatore esegue la chiamata ai sottoprogrammi ADC e STORE. Il sottoprogramma ADC fa acquisire i 10 bit del convertitore e il sottoprogramma STORE memorizza il valore nella memoria a lettura/scrittura. Questi programmi potrebbero essere incorporati nel programma principale del calcolatore, main task.
Il sottoprogramma ADC è già stato incontrato nell'Esempio 2-4, mentre il sottoprogramma STORE non viene riportato dal momento che non sappiamo ancora quali siano i criteri di memorizzazione che vorremo scegliere. Questo infatti sarà deciso nei prossimi esempi.

ESEMPIO 2-8
PROGRAMMA DI TEST SU UN FLAG CHE
UTILIZZA I SOTTOPROGRAMMI ADC E STORE

```
                * 003 126
003 126 333     IN          LEGGE IL FLAG DI TEST
003 127 173     173
003 130 346     ANI         MASCHERA TUTTI GLI ALTRI BIT
003 131 001     001         MASCHERA = 00000001
```

```
003 132 312             JZ        FLAG NON PRESENTE, CONTINUA
003 133 145             CONT
003 134 003             0
003 135 323             OUT       AZZERA IL FLAG DEL FLIP-FLOP
003 136 173             173
003 137 315             CALL      FLAG PRESENTE, ESEGUE LA CONVERSIONE
003 140 000             ADC
003 141 100             0
003 142 315             CALL      MEMORIZZA I DATI UTILIZZANDO
003 143 000             STORE     QUESTA CHIAMATA AL SOTTOPROGRAMMA DI
003 144 200             0         MEMORIZZAZIONE DA AGGIUNGERE AL
                                  PROGRAMMA
003 145 000    CONT,    0         CONTINUA CON QUESTA ISTRUZIONE
                                  SE IL FLAG NON È PRESENTE O DOPO UNA
                                  LETTURA DAL CONVERTITORE
```

Nel secondo esempio di programma di controllo del temporizzatore, Esempio 2-9, un solo sottoprogramma ADCSTR sostituisce i due precedenti ed è completamente *trasparente* dal momento che tutti i registri impiegati sono prima memorizzati nello stack con le istruzioni di PUSH e successivamente recuperati con le istruzioni di POP. Questo sottoprogramma ADCSTR esegue una sola conversione di 10 bit impiegando il convertitore A/D precedentemente interfacciato con il calcolatore. Il valore a 10 bit è in seguito memorizzato nelle locazioni 005 000 e 005 001, senza aprire un file di dati, sebbene ciò possa essere fatto. *Solo l'attuale valore a 10 bit è mantenuto.* Probabilmente il programma MAIN TASK impiegando in qualche modo l'informazione, potrà costituire un file di dati.

ESEMPIO 2-9
TIPICO PROGRAMMA DI TEST SU UN FLAG CHE UTILIZZA I SOTTOPROGRAMMI ADC E STORE

```
                        * 003 126
003 126 333             IN        LEGGE I FLAG DI TEST
003 127 173             173
003 130 346             ANI       MASCHERA TUTTI GLI ALTRI BIT
003 131 001             001       MASCHERA = 00000001
003 132 312             JZ        FLAG PRESENTE, CONTINUA
003 133 142             CONT
003 134 003             0
003 135 323             OUT       AZZERA IL FLAG DI FLIP-FLOP
003 136 173             173
003 137 315             CALL      CHIAMATA AL SOTTOPROGRAMMA CHE
                                  LEGGE E
003 140 000             ADCSTR    MEMORIZZA UN NUOVO VALORE IN MEMORIA
003 141 010             0         ALLA LOCAZIONE CHIAMATA NEWVAL
003 142 000    CONT,    0         CONTINUA CON QUESTA ISTRUZIONE
                                  SE IL FLAG NON È PRESENTE O DOPO
                                  UN'ACQUISIZIONE DEL CONVERTITORE


                        * 010 000
010 000 365    ADCSTR,  PUSHPSW   SALVA IL REGISTRO A E I FLAG
010 001 305             PUSHB     SALVA I REGISTRI B E C
010 002 345             PUSHH     SALVA I REGISTRI H E L
010 003 323             OUT       INIZIA LA CONVERSIONE
010 004 037             037
```

```
010 005 333   CHK,     IN         LEGGE IL BIT DI STATO
010 006 066            066
010 007 306            ADI        TESTA IL FLAG
010 010 200            200        AGGIUNGENDO 1
010 011 322            JNC        SE NON C'È CARRY, ESEGUE DI NUOVO IL TEST
010 012 005            CHK
010 013 010            0
010 014 117            MOVCA      I DUE BIT PIU' SIGNIFICATIVI VENGONO
                                  SALVATI IN C
010 015 333            IN         LEGGE GLI 8 BIT MENO SIGNIFICATIVI
010 016 065            065
010 017 041            LXIH       INIZIALIZZA I PUNTATORI ALLA MEMORIA
010 020 000            NEWVAL
010 021 005            0
010 022 167            MOVMA      MEMORIZZA GLI 8 BIT MENO SIGNIFICATIVI
010 023 043            INXH       INCREMENTA IL PUNTATORE ALLA MEMORIA
010 024 161            MOVMC      MEMORIZZA I 2 BIT PIU' SIGNIFICATIVI
010 025 341            POPH       RIPRISTINA I REGISTRI
010 026 301            POPB
010 027 361            POPPSW
010 030 311            RET        RITORNA AL PROGRAMMA PRINCIPALE


                       * 005 000
005 000 000   NEWVAL,0            INDIRIZZO DELLA CELLA DOVE SONO
                                  MEMORIZZATI GLI 8 LSB
005 001 000            0          INDIRIZZO DELLA CELLA DOVE SONO
                                  MEMORIZZATI I 2 MSB
```

Quando si trova il flag a livello logico 1 e quando il calcolatore inizia l'esecuzione del sottoprogramma ADCSTR, passerà un certo tempo per permettere al calcolatore di completare i passi del sottoprogramma e riprendere l'esecuzione del programma MAIN TASK all'indirizzo CONT + tutto ciò ritarderà l'esecuzione del programma nel suo complesso. In questo caso, il tempo non è significativo, ma è proprio questa incertezza nei tempi di esecuzione del programma da parte del calcolatore che richiede lo sviluppo di un clock in tempo reale.

In entrambi questi esempi di programma (Esempi 2-8 e 2-9) i passi di programma di test del flag e della parte decisionale devono essere aggiunti al MAIN TASK in modo che il temporizzatore possa essere testato e la sua condizione rimanda sotto il controllo del calcolatore.
Se il programma MAIN TASK è eseguito molto velocemente tanto meglio, tuttavia la lunghezza delle operazioni può causare incertezza nel tempo di esecuzione dal momento che è impossibile conoscere quando il calcolatore ritorna dal pezzo di programma che controlla il flag aggiunto.
Nel caso in cui il tempo impiegato nell'esecuzione di questa parte è dell'ordine di 50 o 100 millisecondi, si avrà una lettura della temperatura ogni 10 minuti e 1/10 di secondo; se invece il tempo richiesto è dell'ordine di qualche minuto, le letture avverranno ogni 12, 13 o più minuti e tutto ciò non sarebbe accettabile.

TEMPORIZZATORI DI INTERRUPT

Gli interrupt sembrano rappresentare un'alternativa interessante ai temporizzatori di flag dato che sono serviti immediatamente dal calcolatore. Quando gli interrupt sono usati con un sistema basato sull'8080, sarà vostra responsabilità abilitare l'interrupt e costruire un flag di interrupt nonché una porta in modo che l'istruzione di restart a un singolo byte, RST 7 venga letta dall'8080 non appena si verifica l'interrupt.

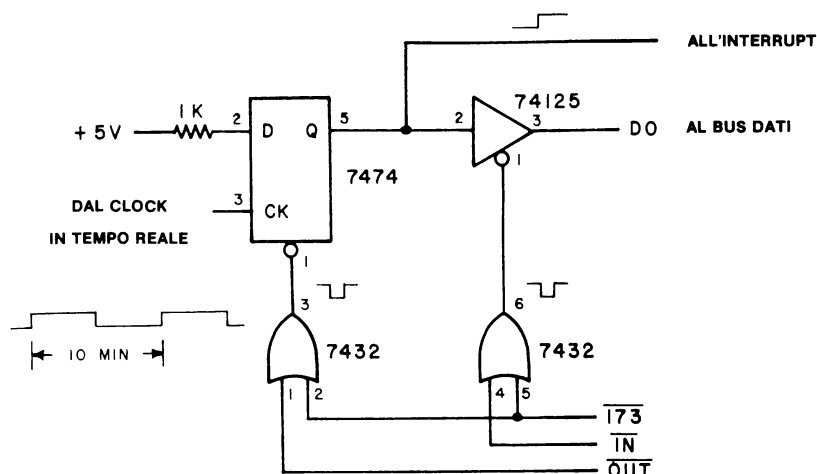



Figura 2-12. Circuito di interrupt. Il flag può essere testato sia mediante un interrupt che mediante un'operazione a controllo di programma.

Daremo per scontato che esiste solo questo unico dispositivo di interrupt. Ricorderete che un'istruzione di restart fa sì che il microcalcolatore 8080 chiami un sottoprogramma in uno speciale indirizzo di vettore. In questo caso ci sarà una chiamata al sottoprogramma nella locazione 000 070. Ciò significa che un'istruzione di reinizio di 377 (RST7) deve essere letta dall'8080 quando si verifica l'interrupt. Ricordate che: *prima di usare l'istruzione di restart, dovete fissare lo stack nella memoria a lettura/scrittura.*

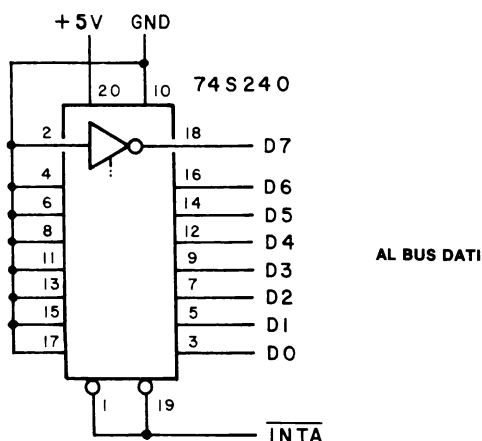



Figura 2-13. La porta di codice dell'interrupt cablata per generare l'istruzione RST7 o 377. Il circuito integrato SN74S240 è usato come dispositivo di ingresso three-state.

Nell'esempio che segue abbiamo usato la stessa circuiteria di flag, usata per il temporizzatore testato a controllo di programma, e che è illustrata nella figura 2-12. Noterete che la sola differenza tra questo ciclo e il circuito di (Figura 2-11) è l'aggiunta di un segnale dell'interrupt, TO INTERRUPT, che è collegato al piedino di ingresso di interrupt del chip del microprocessore 8080, cioé il piedino 14. La Figura 2-13 contiene un esempio di porta per il codice di interrupt. Per fornire l'istruzione 377 (RST7), quando il buffer è abilitato con il segnale di (?) abilitazione all'interrupt, è stato usato un buffer three-state a otto bit SN74S240.

Un fronte positivo applicato all'input di clock del flip-flop tipo D causerà a questo punto un interrupt che sarà servito o riconosciuto dall'8080 non appena ha terminato l'esecuzione dell'operazione corrente. Il tempo di risposta in questi casi è dell'ordine di pochi microsecondi. Quando l'8080 riconosce l'interrupt, l'istruzione RST7 viene campionata sul bus dati e nel *registro di istruzione* dell'8080 deve essere decodificata per far eseguire dall'8080 la chiamata al sottoprogramma che inizia all'indirizzo 000 070, là dove comincia anche il programma di servizio del convertitore analogico/digitale.

Il sottoprogramma di servizio dell'interrupt illustrato nell'Esempio 2-10 presenta alcune caratteristiche su cui vale la pena soffermarsi. Notate che il convertitore A/D viene fatto partire prima che vengano eseguite le due istruzioni PUSH.
Poiché l'istruzione OUT non modifica nessuno dei registri dell'8080, ciò è certamente valido. Iniziare la conversione prima fa risparmiare tempo dato che il convertitore la eseguirà mentre vengono eseguite le due istruzioni di PUSH. Quando si usa un 8080 con un periodo di clock di 500 nanosecondi, si possono risparmiare 11 microsecondi in modo che la conversione A/D è quasi completata se si sta usando un convertitore A/D veloce ad approssimazioni successive, come il modello AD 7570 della Analog Devices.
L'istruzione LHLD carica la coppia di registri H con l'indirizzo che punta alle locazioni di memorizzazione dei dati. Ciò aggiunge altri otto microsecondi al tempo che intercorre tra l'inizio della conversione e i passi di controllo del flag di fine-conversione.

Se tra il comando usato per far partire il processo di conversione e i comandi usati per controllare lo stato del flag DONE/$\overline{\text{BUSY}}$ è stato collocato un numero sufficiente di istruzioni, il controllo del flag può non essere necessario.
Il convertitore completerà il processo di conversione nel tempo in cui il calcolatore sarà pronto per il caricamento dei dati. In questo caso sarete in grado di fare a meno dei passi di controllo del flag e potrete caricare i dati direttamente. Questa tecnica dovrebbe essere usata solo quando si sa con precisione quanto è lungo il processo di conversione e se tale durata è più breve del tempo necessario al calcolatore per eseguire le operazioni "overhead" di supervisione.

Il software illustrato nell'Esempio 2-10 continua a caricare e a memorizzare i valori della temperatura ogni 10 minuti fino a quando non verrà disattivato. Si può anche aggiungere dell'altro software per contare i 144 punti durante 24 ore.
Oppure si può utilizzare per questo il programma MAIN TASK, che può accedere e usare i valori di indirizzo memorizzati su POINT per calcolare quanti punti sono stati caricati e memorizzati. Il software per il conteggio dei punti può anche essere aggiunto al sottoprogramma di servizio del convertitore A/D, ADCSVC. Il programma può essere disabilitato, disattivando semplicemente il flag di abilitazione dell'interrupt, con un istruzione di disattivazione dell'interrupt, DI, 363. È questo un punto molto importante. Per implementare questa soluzione basata su interrupt del problema sulla temporizzazione a 10 minuti in MAIN TASK richiede solo un'istruzione di abilitazione dell'interrupt e un'istruzione di caricamento del puntatore dello stack e non c'è nessun software per controllare un flag o per ritardare l'esecuzione del MAIN TASK.

Il secondo esempio di software dell'interrupt (Esempio 2-11) è completo, dato che contiene istruzioni usate per realizzare un file di dati a 144 punti e il programma quindi salterà all'etichetta DONE quando sono stati acquisiti tutti i punti.

ESEMPIO 2-10

```
                        * 000 070
000 070 323   ADCSVC,  OUT        FA PARTIRE IL CONVERTITORE
000 071 037            037
000 072 365            PUSHPSW    SALVA IL REGISTRO A E I FLAG
000 073 345            PUSHH      SALVA I REGISTRI H E L
000 074 052            LHLD       METTE I PUNTATORI ALLA MEMORIA IN H E L
000 075 000            POINT      IN MODO CHE I DATI POSSANO ESSERE
                                  MEMORIZZATI
000 076 120            0
000 077 333   TEST,    IN         LEGGE IL BIT DI FLAG
000 100 066            066
000 101 306            ADI        AGGIUNGE UN 1 AL BIT DI FLAG
000 102 200            200
000 103 322            JNC        È UN 1?
000 104 077            TEST       NO, NON ANCORA ESEGUITO
000 105 000            0
000 106 107            MOVBA      SI, ADC ESEGUITO, MEMORIZZA 2 MSB IN B
000 107 333            IN         CARICA 8 LSB
000 110 065            065
000 111 167            MOVMA      LI TRASFERISCE IN MEMORIA
000 112 043            INXH       INCREMENTA IL PUNTATORE ALLA MEMORIA
000 113 160            MOVMB      CARICA I DUE MSB IN MEMORIA
000 114 043            INXH       INCREMENTA DI NUOVO IL PUNTATORE ALLA
                                  MEMORIA
000 115 042            SHLD       SALVA L'INDIRIZZO DELL'AREA DI MEMORIA
000 116 000            POINT
000 117 120            0
000 120 341            POPH       RIPRISTINA I REGISTRI H E L
000 121 361            POPPSW     RIPRISTINA IL REGISTRO A E I FLAG
000 122 323            OUT        AZZERA IL FLAG DI INTERRUPT
000 123 173            173
000 124 373            EI         RIABILITA L'INTERRUPT
000 125 311            RET        RITORNA AL PROGRAMMA PRINCIPALE

                        * 120 000
120 000 000   POINT,   000        QUESTO È IL PUNTO IN CUI È TENUTO
                                  L'INDIRIZZO DELL'AREA
120 001 020            020        DI MEMORIA ADC
                                  L'AREA DI MEMORIA DATI INIZIA
                                  ALL'INDIRIZZO 020 000. POTETE SISTEMARE IL
                                  VOSTRO
                                  INDIRIZZO DEL PUNTATORE QUI, MA QUESTE
                                  DUE LOCAZIONE DEVONO ESSERE NELLA
                                  MEMORIA A LETTURA/SCRITTURA
```

ESEMPIO 2-11
QUESTO È IL PROGRAMMA SERVIZIO DELL'INTERRUPT ADC
CON I PASSI IN PIU' PER UN CONTATORE DI PUNTI E
UN'USCITA VERSO LA ROUTINE DI "DONE"
QUANDO TUTTI I PUNTI SONO STATI ACQUISITI

```
                        * 000 070
000 070 323   ADCSVC,  OUT        FA PARTIRE IL CONVERTITORE
000 071 037            037
000 072 365            PUSHPSW    SALVA IL REGISTRO A E I FLAG
```

```
000 073 325            PUSHD    SALVA I REGISTRI D E E
000 074 305            PUSHB    SALVA I REGISTRI B E C
000 075 345            PUSHH    SALVA I REGISTRI H E L
000 076 052            LHLD     PRENDE IL CONTATORE DI PUNTI
000 077 002            COUNT
000 100 120            0
000 101 353            XCHG     SCAMBIA LA COPPIA DI REGISTRI D E H
000 102 052            LHLD     METTE I PUNTATORI ALLA MEMORIA IN H E L
000 103 000            POINT    IN MODO CHE I DATI POSSANO ESSERE
                                MEMORIZZATI
000 104 120            0
000 105 333   TEST,    IN       LEGGE IL BIT DI FLAG
000 106 066            066
000 107 306            ADI      AGGIUNGE UN 1 AL BIT DI FLAG
000 110 200            200
000 111 322            JNC      È UN 1?
000 112 105            TEST     NO, NON ANCORA ESEGUITO
000 113 000            0
000 114 107            MOVBA    SI, ADC ESEGUITO, MEMORIZZA 2 MSB IN B
000 115 333            IN       LEGGE GLI OTTO BIT PIU' SIGNIFICATIVI
000 116 065            065
000 117 167            MOVMA    LI CARICA IN MEMORIA
000 120 043            INXH     INCREMENTA IL PUNTATORE ALLA MEMORIA
000 121 160            MOVMB    CARICA I DUE BYTE PIU' SIGNIFICATIVI IN
                                MEMORIA
000 122 043            INXH     INCREMENTA DI NUOVO IL PUNTATORE ALLA
                                MEMORIA
000 123 025            DCRD     DECREMENTA IL CONTATORE DI PUNTI
000 124 312            JZ       SONO STATI PRESI TUTTI I PUNTI?
000 125 146            EXIT     SI, VAI IN USCITA
000 126 000            0
000 127 042            SHLD     SALVA L'INDIRIZZO DELL'AREA DI MEMORIA
                                DATI
000 130 000            POINT
000 131 120            0
000 132 353            XCHG     SCAMBIA LA COPPIA DEI REGISTRI D, E CON H,
                                L
000 133 042            SHLD     SALVA IL CONTATORE DI PUNTI
000 134 002            COUNT
000 135 120            0
000 136 341            POPH     RIPRISTINA I REGISTRI H E L
000 137 301            POPB     RIPRISTINA I REGISTRI B E C
000 140 321            POPD     RIPRISTINA I REGISTRI D E E
000 141 361            POPPSW   RIPRISTINA IL REGISTRO A E I FLAG
000 142 323            OUT      CANCELLA IL FLAG DI INTERRUPT
000 143 173            173
000 144 373            EI       RIATTIVA L'INTERRUPT
000 145 311            RET      RITORNA AL PROGRAMMA PRINCIPALE.
000 146 341   EXIT,    POPH     RIPRISTINA I REGISTRI
000 147 301            POPB
000 150 321            POPD
000 151 361            POPPSW
000 152 063            INXSP    INCREMENTA IL PUNTATORE DI STACK
000 153 063            INXSP    OLTRE L'INDIRIZZO DI RITORNO
000 154 000   DONE,    0        SPAZIO PER I PROGRAMMI ADDIZIONALI
                                DA QUI IN AVANTI
```

		* 120 000	
120 000 000	POINT,	000	QUI È DOVE È MANTENUTA DELL'AREA DI MEMORIA
120 001 020		020	PER L'ADC. IN QUESTO PROGRAMMA L'AREA INIZIA ALL'INDIRIZZO 020 000. POTETE SISTEMARE IL VOSTRO INDIRIZZO DAL PUNTATORE QUI, MA QUESTE DUE LOCAZIONE DEVONO ESSERE NELLA MEMORIA A LETTURA/ SCRITTURA
120 002 000	COUNT,	000	IL CONTATORE DI PUNTI È MEMORIZZATO QUI
120 003 220		220	AL SECONDO BYTE, MENTRE IL PRIMO NON È UTILIZZATO, 220 OTTALE = 144 DECIMALE

Questo programma è leggermente più complicato di quello riportato nell'Esempio 2-10 in quanto usciamo dal sottoprogramma e non abbiamo intenzione di utilizzare l'indirizzo di ritorno memorizzato nello stack durante l'esecuzione da parte del calcolatore dell'istruzione di restart o di chiamata. Il doppio incremento del puntatore di stack serve a spostare il puntatore oltre gli indirizzi di memoria impiegati per la memorizzazione dei due byte dell'indirizzo di ritorno.
Nel caso si voglia utilizzare l'indirizzo di ritorno, è sufficiente semplicemente sostituire le due istruzioni INXSP con due istruzioni NO8.

L'area di memoria dall'indirizzo DONE è stata lasciata libera in modo da permettere l'aggiunta di eventuali routine per emettere o visualizzare le 144 letture di temperatura. Noterete che saltando all'istruzione DONE, probabilmente cessate le operazioni del programma MAIN TASK.

I clock in tempo reale sono dispositivi periferici molto utili quando si vuole avere su un piccolo calcolatore una temporizzazione accurata. I clock possono essere sia semplici come l'oscillatore e la catena di contatori che abbiamo proposto precedentemente, sia dispositivi più complessi come il circuito integrato INTEL 8253 temporizzatore programmabile oppure il controllore multifunzione TMS-5501 della Texas Instruments; ciascuno di questi dispositivi comprende dei registri programmabili che possono essere caricati con un particolare valore. Dopo che è stato effettuato il numero di conteggi prefissato, il dispositivo può sia generare un interrupt che alzare un flag che può essere testato con accorgimenti software. La frequenza di conteggio è determinata da un oscillatore al quarzo esterno.

UTILIZZAZIONE DEI DATI

Una volta che i dati acquisiti dai sensori di temperatura sono stati memorizzati dal calcolatore, si deve decidere che uso farne. Si potrebbe pensare di emettere i dati su una telescrivente per un'eventuale supervisione, ma potrebbe essere poco significativo dal momento che i dati sono memorizzati come valori di tensione a 10 bit e non come temperatura. Si potrebbe anche pensare di aggiungere routine per convertire i dati a 10 bit in temperature, ma un formato di uscita più semplice è possibile.

Impiegheremo un convertitore D/A a 10 bit per emettere i dati su un piccolo oscilloscopio per la visualizzazione dopo un periodo di acquisizione di 24 ore. Ciò può fornire all'utente un profilo dell'andamento della temperatura delle ultime 24 ore. Lo stesso tipo di formato può essere emesso su un registratore di carta per realizzare un tipo di visualizzazione permanente.

Il programma da utilizzare per la visualizzazione dei valori a 10 bit è simile a quello dell'Esempio 1-18. Nel prossimo esempio impiegheremo un programma di ritardo di 10

minuti in quanto semplifica l'aggiunta del programma, così come semplifica l'esempio il non impiego, per altro possibile, di procedure di interrupt o di temporizzatori di flag. Il programma riporta gli stessi passi dell'Esempio 2-7, eccetto la parte relativa alla visualizzazione che è stata aggiunta al posto dell'istruzione di HLT all'etichetta DONE. Riportiamo di seguito il programma completo:

ESEMPIO 2-12
PROGRAMMA PER L'ACQUISIZIONE E LA VISUALIZZAZIONE DI 144 PUNTI

```
                         *003 000
003 000 061   START,     LXISP     CARICA IL PUNTATORE DI STACK PER I
                                   SOTTOPROGRAMMI
003 001 377              377
003 002 003              003
003 003 026              MVID      CARICA D COME IL CONTATORE DEI 144 PUNTI
003 004 220              220       220 OTTALE= 144 DECIMALE
003 005 041              LXIH      CARICA I REGISTRI H E L COME PUNTATORI
                                   ALLA MEMORIA
003 006 000              000
003 007 002              002
003 010 323   CONVRT,    OUT       INIZIA LA CONVERSIONE
003 011 037              037
003 012 333   TEST,      IN        LEGGE IL BIT DI FLAG
003 013 066              066
003 014 306              ADI       AGGIUNGE 1 AL BIT DI FLAG PER CAUSARE
003 015 200              200       EVENTUALMENTE UN RIPORTO
003 016 322              JNC       NESSUN RIPORTO, ESEGUE DI NUOVO IL TEST.
003 017 012              TEST
003 020 003              0
003 021 107              MOVBA     SALVA I BIT PIU' SIGNIFICATIVI
003 022 333              IN        LEGGE GLI 8 BIT MENO SIGNIFICATIVI
003 023 065              065
003 024 167              MOVMA     LI MEMORIZZA

003 104 220              220
003 105 176   LOOP,      MOVAM     PRENDE GLI 8 BIT MENO SIGNIFICATIVI
003 106 323              OUT       E LI EMETTE SUL DAC
003 107 054              054
003 110 043              INXH      INCREMENTA IL PUNTATORE ALLA MEMORIA
003 111 176              MOVAM     PRENDE I DUE BIT PIU' SIGNIFICATIVI
003 112 323              OUT       ED EMETTE ANCHE QUESTI
003 113 055              055
003 114 323              OUT       EMETTE L'IMPULSO DI STROBE PER LA DOPPIA
                                   BUFFERIZZAZIONE
003 115 056              056       AL MODULO DAC
003 116 043              INXH      INCREMENTA DI NUOVO IL PUNTATORE
003 117 025              DCRD      DECREMENTA IL CONTATORE DI PUNTI.
003 120 312              JZ        È= 0?
003 121 100              DONE      SI, INIZIALIZZA DI NUOVO E VISUALIZZA
003 122 003              0         ANCORA IL FILE
003 123 303              JMP       NO, PROSEGUE
003 124 105              LOOP
003 125 003              0
003 025 043              INXH      INCREMENTA IL PUNTATORE ALLA MEMORIA
003 026 160              MOVMB     MEMORIZZA I 2 BIT PIU' SIGNIFICATIVI
003 027 043              INXH
```

```
003 030 005             DCRB      DECREMENTA IL CONTATORE DI PUNTI
003 031 312             JZ        È= 0?
003 032 100             DONE      SI, ESEGUITO
003 033 003             0
003 034 315             CALL      ESEGUE IL RITARDO
003 035 042             DELAY
003 036 003             0
003 037 303             JMP       RITORNA ED ESEGUE DI NUOVO.
003 040 010             CONVRT
003 041 003             0
                        QUI È DOVE DEVE ESSERE POSTO
                        IL SOTTOPROGRAMMA DI RITARDO DI 10 PUNTI.
003 042 000   DELAY,    0

                        IL PROGRAMMA DI VISUALIZZAZIONE SOSTITUISCE
                        L'ISTRUZIONE DI HALT

                        * 003 100
003 100 041   DONE,     LXIH      INIZIALIZZA IL PUNTATORE ALLA MEMORIA
003 101 000             000
003 102 002             002
003 103 026             MVID      INIZIALIZZA IL CONTATORE DI PUNTI.
```

Questo programma risolve brillantemente la necessità di visualizzare i dati relativi al periodo di 24 ore precedenti, ma esiste un limite alla sua utilità in quanto i nuovi valori non vengono visualizzati prima che siano trascorse le 24 ore e tutti i 144 punti siano stati acquisiti. È possibile invece che il programma sia modificato in modo da visualizzare i dati non solo al termine del periodo di 24 ore di acquisizione, ma in modo da mantenere aggiornata l'immagine delle ultime 24 ore. La Figura 2-14 illustra come si comporta un'immagine continuamente aggiornata.

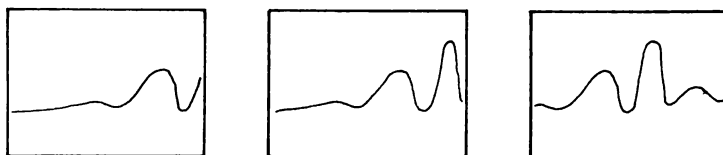
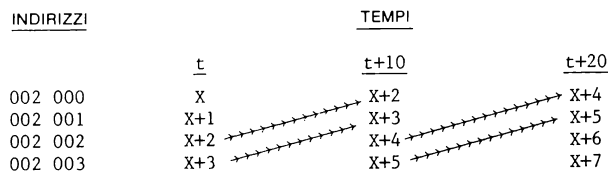



Figura 2-14. Un'immagine aggiornata mostrata in tre tempi, t0, t1, t2. I dati sembrano scorrere da destra a sinistra.

Anche se queste riproduzioni di aggiornamento sono esagerate, esse illustrano che cosa sia richiesto all'hardware e al software. I nuovi punti di temperatura vengono aggiunti all'immagine sul lato destro al momento in cui vengono aggiunti e le letture vecchie escono dal lato sinistro dello schermo, in modo da mantenere l'informazione riguardante l'ultimo *periodo* di 24 ore. Il movimento dei dati nell'area di memoria dedicata alla visualizzazione è illustrata graficamente qui sotto:

INDIRIZZI	TEMPI		
	t	t+10	t+20
002 000	X	X+2	X+4
002 001	X+1	X+3	X+5
002 002	X+2	X+4	X+6
002 003	X+3	X+5	X+7

È da evidenziare il fatto che i valori relativi a X e X +1 sono entrambi scomparsi al tempo t +10 poiché ciascun valore di temperatura è di 10 bit e richiede quindi due byte per la sua memorizzazione.

I nuovi punti sono aggiunti alla fine del file dei dati della temperatura e dal momento che il fascio dell'oscilloscopio si sposta da sinistra a destra, gli ultimi valori che sono acquisiti sono visualizzati alla destra dello schermo. Il programma necessario per spostare i dati in giù di due posizioni è riportato nell'Esempio 2-13. È necessario lo spostamento di soli 143 punti in quanto il 144 è dato dall'ultima lettura e sarà aggiunto alla fine del file. Tuttavia lo spostamento di tutti i 144 non è per niente diverso.

ESEMPIO 2-13
QUESTO SOTTOPROGRAMMA TRASFERISCE I VALORI DEI DATI IN DUE INDIRIZZI DELLA MEMORIA A LETTURA/ SCRITTURA

```
                          * 003 000
003 000 052   UPDATE,  LHLD      PRENDE L'INDIRIZZO DI FILE
003 001 100            POINT
003 002 003            0
003 003 072            LDA       PRENDE IL CONTATORE DI PUNTI
003 004 102            COUNT
003 005 003            0
003 006 137            MOVEA     LO CARICA NEL REGISTRO E
003 007 043            INXH      INCREMENTA L'INDIRIZZO DUE VOLTE
003 010 043            INXH
003 011 006   NXTPNT,  MVIB      REGOLA B= BYTE PER PUNTO
003 012 002            002
003 013 116   BYTE2,   MOVCM     PRENDE I DATI
003 014 053            DCXH      DECREMENTA L'INDIRIZZO DUE VOLTE
003 015 053            DCXH
003 016 161            MOVMC     VI METTE I DATI
003 017 043            INXH      SPOSTA L'INDIRIZZO FINO A TRE
003 020 043            INXH
003 021 043            INXH
003 022 005            DCRB      TUTTI I BYTE SONO STATI ESEGUITI?
003 023 302            JNZ       NO, ESEGUE UN ALTRO TRASFERIMENTO
003 024 013            BYTE2
003 025 003            0
003 026 035            DCRE      SI, DECREMENTA IL CONTATORE DI PUNTI
003 027 302            JNZ       ALTRI PUNTI?
003 030 011            NXTPNT    SI, ESEGUE UN ALRO TRASFERIMENTO
003 031 003            0
003 032 311            RET       NO, RITORNA

                       * 003 100
003 100 000   POINT,   0         IL PUNTATORE ALL'INDIRIZZO È
                                 MEMORIZZATO QUI
003 101 000            0         IN QUESTO ESEMPIO
003 102 000   COUNT,   0         IL CONTATORE DI PUNTI È MEMORIZZATO QUI
```

Se dobbiamo usare il calcolatore per visualizzare in modo continuo i dati relativi alla temperatura e aggiornarli non appena vengono acquisiti nuovi valori è ovvio che il calcolatore non può essere usato anche per eseguire le operazioni del sottoprogrammma di ritardo da 10 minuti, che abbiamo reintrodotto nell'Esempio 2-12.

Non intendiamo trattare i passi relativi alla visualizzazione come programma MAIN TASK e useremo un temporizzatore di interrupt per segnalare la fine di ogni periodo da 10 minuti. Il software del convertitore A/D e il sottoprogramma di aggiornamento del file (Esempio 2-13) farà parte del sottoprogramma di servizio dell'interrupt.

I due schemi a blocchi nelle Figure 2-15 e 2-16 illustrano l'esecuzione del programma del display e dell'interrupt. L'interrupt si può verificare in qualsiasi momento durante l'esecuzione del software del display, per questo gli schemi a blocchi non mostrano il collegamento diretto con il sottoprogramma di servizio dell'interrupt, INTSVC.

I principali programmi di funzionamento illustrano gli ultimi 144 valori della temperatura memorizzati nel file di dati. Il convertitore acquisisce nuovi valori e il file viene aggiornato solo quando il temporizzatore di interrupt segnala che è trascorso un periodo di 10 minuti. Inizialmente i dati nel file verranno messi a zero, mediante le istruzioni di software CLEAR= ciò produrrà una linea diritta quando il dato viene visualizzato dall'oscilloscopio in modo che non si avranno letture non corrette delle temperature fornite dai"dati casuali" che si trovano nella memoria a lettura/scrittura del calcolatore al momento dell'accensione.

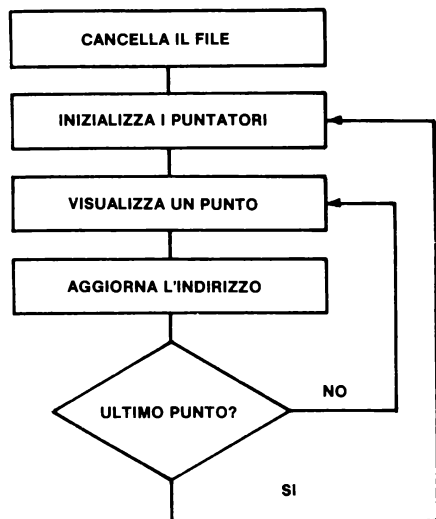



Figura 2-15. Il programma MAIN TASK visualizza tutti i 144 valori di temperatura.

Figura 2-16. Programma di manipolazione del convertitore basato su interrupt e del file di dati.

```
                              ESEMPIO 2-14
                              QUESTO È UN PROGRAMMA COMPLETO DI ACQUISIZIONE
                              DATI BASATO SU INTERRUPT CON IL SOFTWARE DI
                              VISUALIZZAZIONE USATO COME "MAIN TASK"

                              * 003 000
003 000 061     START,        LXISP        CARICA IL PUNTATORE DI STACK
003 001 370                   370
003 002 004                   004
003 003 373                   EI           ABILITA L'INTERRUPT
```

```
003 004 052            LHLD       PRENDE L'INDIRIZZO DEL FILE DI DATI
003 005 054            POINT
003 006 003            0
003 007 072            LDA        PRENDE IL CONTATORE DI PUNTI
003 010 060            COUNT
003 011 003            0
003 012 137            MOVEA      LO SPOSTA NEL REGISTRO E
003 013 257  CLEAR,    XRAA       CANCELLA A
003 014 167            MOVMA      LO MEMORIZZA
003 015 043            INXH       INCREMENTA IL PUNTATORE
003 016 167            MOVMA      LO RIPETE
003 017 043            INXH
003 020 035            DCRE       DECREMENTA IL CONTATORE
003 021 302            JNZ        SE IL CONTATORE NON È 0
003 022 013            CLEAR      CANCELLA LA LOCAZIONE SUCCESSIVA
003 023 003            0
003 024 052  DISPL,    LHLD       PRENDE L'INDIRIZZO DEL FILE
003 025 054            POINT
003 026 003            0
003 027 072            LDA        PRENDE IL CONTATORE DI PUNTI
003 030 060            COUNT
003 031 003            0
003 032 137            MOVEA      LO SPOSTA NEL REGISTRO E
003 033 176  MORE,     MOVAM      PRENDE GLI 8 BIT MENO SIGNIFICATIVI
003 034 323            OUT        USCITA AL DAC
003 035 054            054
003 036 043            INXH       INCREMENTA IL PUNTATORE ALL'INDIRIZZO
003 037 176            MOVAM      PRENDE I DUE BIT PIU' SIGNIFICATIVI DI UN
                                  PUNTO
003 040 323            OUT        USCITA AL DAC
003 041 055            055
003 042 323            OUT        DECREMENTA IL CONTATORE
003 043 056            056
003 044 043            INXH       INCREMENTA DI NUOVO L'INDIRIZZO
003 045 035            DCRE       DECREMENTA IL CONTATORE
003 046 302            JNZ        ALTRI PUNTI DA METTERE IN USCITA?
003 047 033            MORE       SI,
003 050 003            0
003 051 303            JMP        NO, VISUALIZZA TUTTO DI NUOVO
003 052 024            DISPL
003 053 003            0

003 054 000  POINT,    000        PARTE BASSA DELL'INDIRIZZO DEL FILE DI
                                  DATI
003 055 020            020        PARTE ALTA DELL'INDIRIZZO DEL FILE DI DATI
003 056 036  NEWPNT,036           PRIMO INDIRIZZO DELL'ULTIMO

003 057 021            021
003 060 220  COUNT,    220        NUMERI DI PUNTI DATO, FINO A
003 061 000            000        377 = 255 DECIMALE
                                  220 = 144 DECIMALE

                                  QUESTO E' UN SOTTOPROGRAMMA ADC, CHE
                                  FA PARTIRE UNA CONVERSIONE, ASPETTA
                                  L'ESECUZIONE DEL FLAG E MEMORIZZA I DATI
                                  NELL'ULTIMA LOCAZIONE DEL FILE
003 062 323  CONVRT,OUT           FA PARTIRE UNA CONVERSIONE
003 063 037            037
003 064 333  CHK,      IN         LEGGE IL BIT DI FLAG
003 065 066            066
```

```
003 066 306              ADI        AGGIUNGE AD ESSO UN 1
003 067 200              200
003 070 322              JNC        SE NON C'E' RIPORTO, CONTROLLA DI NUOVO
003 071 064              CHK
003 072 003              0
003 073 117              MOVCA      MEMORIZZA I 2 BIT PIU' SIGNIFICATIVI
003 074 333              IN
003 075 065              065        LEGGE GLI 8 BIT MENO SIGNIFICATIVI
003 076 052              LHLD       CARICA L'INDIRIZZO DEL
003 077 056              NEWPNT     NUOVO O ULTIMO PUNTO
003 100 003              0
003 101 167              MOVMA      MEMORIZZA GLI 8 BIT MENO SIGNIFICATIVI
003 102 043              INXH       INCREMENTA L'INDIRIZZO
003 103 161              MOVMC      MEMORIZZA I DUE BIT PIU' SIGNIFICATIVI
003 104 311              RET

                                    QUESTO SOTTOPROGRAMMA SPOSTA I
                                    VALORI DEI DATI IN DUE INDIRIZZI DELLA
                                    MEMORIA A LETTURA/SCRITTURA

003 105 052    UPDATE,   LHLD       PRENDE L'INDIRIZZO DEL FILE
003 106 054              POINT
003 107 003              0
003 110 072              LDA        PRENDE IL CONTATORE DI PUNTI
003 111 060              COUNT
003 112 003              0
003 113 137              MOVEA      LO MEMORIZZA NEL REGISTRO E
003 114 043              INXH       INCREMENTA L'INDIRIZZO DUE VOLTE
003 115 043              INXH
003 116 006    NXTPNT,   MVIB       REGOLA B = ≠ BYTE PER PUNTO
003 117 002              002
003 120 116    BYTE 2,   MOVCM      PRENDE IL DATO
003 121 053              DCXH       DECREMENTA L'INDIRIZZO DUE VOLTE
003 122 053              DCXH
003 123 161              MOVMC      VI METTE I DATI
003 124 043              INXH       SPOSTA L'INDIRIZZO FINO A 3
003 125 043              INXH
003 126 043              INXH
003 127 005              DCRB       TUTTI I BYTE ESEGUITI?
003 130 302              JNZ        NO, ESEGUE UN ALTRO TRASFERIMENTO

003 131 120              BYTE2
003 132 003              0
003 133 035              DCRE       SI DECREMENTA IL CONTATORE DI PUNTI
003 134 302              JNZ        ALTRI PUNTI
003 135 116              NXTPNT     SI, NE ESEGUE UN ALTRO
003 136 003              0
003 137 311              RET        NO, RITORNA

                                    SOTTOPROGRAMMA DI SERVIZIO DI
                                    INTERRUPT PER IL CLOCK IN TEMPO REALE


                         * 000 070
000 070 323    INTSVC,   OUT        CANCELLA IL FLAG DI INTERRUPT
000 071 173              173
000 072 365              PUSHPSW    SALVA IL FLAG E I REGISTRI
000 073 305              PUSHB
000 074 325              PUSHD
000 075 345              PUSHH
000 076 315              CALL       SPOSTA I DATI NEL PUNTATORE
```

```
000 077 105    UPDATE
000 100 003    0
000 101 315    CALL      ESEGUE UNA CONVERSIONE
000 102 062    CONVRT
000 103 003    0
000 104 341    POPH      REGISTRA I REGISTRI
000 105 321    POPD
000 106 301    POPB
000 107 361    POPPSW
000 110 373    EI        RIABILITA L'INTERRUPT
000 111 311    RET       RITORNO AL PROGRAMMA MAIN TASK
```

Il tempo che occorre per servire il dispositivo di interruzione in questo caso il convertitore A/D, sarà piuttosto breve dato che si usa un convertitore veloce ad approssimazioni successive. Per altri convertitori più lenti, come il metro pannello digitale, che possono impiegare 0,1 secondi, il tempo trascorso nella parte del servizio dell'interrupt del programma sarà più lungo. Per evitare di perdere troppo tempo nel software di servizio dell'interrupt, quindi fuori dal programma MAIN TASK, si può usare uno schema di interrupt a più livelli o di priorità.

Quando si usa un convertitore o un altro dispositivo di interruzione con un tempo di risposta lungo, si possono usare due interrupt. Nel caso di un convertitore A/D lento, un interrupt viene provocato dal clock e l'altro dal convertitore A/D non appena quest'ultimo ha eseguito una conversione. L'interrupt provocato dal clock a intervalli di 10 minuti vettorizza o punta (?) il calcolatore verso un sottoprogramma che fa partire il convertitore A/D, cancella il flag di interrupt del clock, riabilita l'interrupt e ritorna al programma MAIN TASK, che visualizza i dati contenuti nel file. Quando il convertitore A/D lento ha completato la conversione, interrompe l'8080 ma con un'istruzione di reinizio diversa, cioè RST6. Il sottoprogramma di servizio dell'interrupt, puntato (?) dall'istruzione RST6, parte dall'indirizzo 000 060 e viene usato per aggiornare il file di dati, leggere i dati provenienti dal convertitore A/D e poi memorizzarli nel file. In questo modo il calcolatore può utilizzare efficacemente il tempo durante il quale la conversione viene effettuata per altre operazioni. La sola cosa che diamo per scontata è che il convertitore finirà la conversione in corso prima che quella successiva venga richiesta dal tempo di interruzione. Per questa ragione non potremo usare un convertitore con un tempo di conversione di 0,1 secondi assieme ad un sistema di acquisizione dati che richieda più di 10 punti al secondo. Il sistema usato nell'esempio della serra avrà quindi periodi di 10 minuti tra una conversione e l'altra in modo da poter ammettere anche l'impiego di convertitori A/D lenti.

Vi ricordiamo che l'uso degli interrupt è stato ampiamente illustrato nel *Bugbook VI*, Capitolo 23.

La Figura 2-17 contiene un circuito per due interrupt. Mentre il flag DONE/$\overline{\text{BUSY}}$ del convertitore A/D genera un interrupt e l'istruzione RST6, il clock genera un interrupt e l'istruzione RST7. Ricordate che l'istruzione RST7 corrisponde a 377 mentre l'istruzione RST6 corrisponde a 367. Lo schema che segue non si riferisce ad un interrupt.
Il tempo impiegato per la gestione dei dispositivi di interrupt, in questo caso costituiti dal convertitore A/D, è sempre minore di qualsiasi conversione ad approssimazioni successive .

Per altro altri convertitori possono richiedere più tempo, come ad esempio un pannello digitale di misura che può richiedere 0,1 secondi, facendo quindi diventare il tempo dedicato al servizio dell'interrupt una parte considerevole del programma. Per evitare di impiegare periodi di tempo lunghi nella gestione di un interrupt, e questo fuori dal programma MAIN TASK si può utilizzare uno schema di interrupt a più livelli o a priorità. Quando si impiega un convertitore o un qualsiasi altro dispositivo che lavori in interrupt e che richieda un tempo di gestione lungo, si possono utilizzare due interrupt. Nel caso della conversione A/D lenta, un interrupt potrebbe essere provocato dal clock e l'altro dal

convertitore A/D al termine della conversione. Il primo interrupt causato dal clock ogni dieci minuti, provoca un vettore che porta l'esecuzione da parte del calcolatore di un sottoprogramma che fa partire la conversione, azzera il flag di richiesta dell'interrupt e ritorna nuovamente al programma MAIN TASK, che visualizza i dati del file. Quando la conversione è completata, si dovrà generare una nuova e diversa richiesta di interrupt, utilizzando, per esempio, l'istruzione di RST 6 che provoca il salto all'inizio del programma di servizio a partire dall'indirizzo 000 060. Questo sottoprogramma realizzerà l'aggiornamento del file di dati, introducendo il nuovo dato acquisito dal convertitore.

In questo modo il calcolatore può effettivamente eseguire altre funzioni durante il tempo in cui viene effettuata la conversione. L'unico presupposto che è necessario per un corretto funzionamento è che ogni conversione termini prima della richiesta della successiva tramite interrupt. In questo modo non potremo usare dei convertitori con tempo di conversione di 0,1 secondi in un sistema di acquisizione dati che richiede più di 10 acquisizioni al secondo. Il sistema di acquisizione della serra di cui abbiamo discusso comporta un'acquisizione ogni 10 minuti, il che permette l'impiego di convertitori anche estremamente lenti. Le tecniche di uso degli interrupt sono state descritte in maggior dettaglio nel Capitolo 23 del *Bugbook VI*.
Un circuito per la gestione di due interrupt è riportato nella Figura 2-17. Il clock genererà un interrupt e l'istruzione di RST 7 mentre il flag DONE/$\overline{BUSY}$ genererà un secondo interrupt e l'istruizione RST 6. Questo schema non comporta nessuna priorità ed abbiamo aggiunto che il clock e il convertitore non facciano una richiesta simultanea di interrupt.
Il programma necessario per implementare il sistema descritto precedentemente è riportato nell'Esempio 2-15. I programmi di visualizzazione, il MAIN TASK e quello di aggiornamento dei dati non subiscono nessuna variazione dal fatto di adottare il sistema a due interrupt.
La routine di servizio del clock, CKSVC, sostituisce quello di servizio INTSVC, riportata nell'Esempio 2-14, mentre quella relativa all'interruzione dal convertitore, ADSVC, è nuova. Noterete che il nuovo sottoprogramma ADSVC non contiene i passi di programma relativi al test del flag. Perchè? Il flag DONE relativo al convertitore genererà ora l'istruzione di RST 6, in modo da provocare l'esecuzione da parte del calcolatore del sottoprogramma ADSVC ogni volta che viene completata una conversione.

ESEMPIO 2-15
SOTTOPROGRAMMI DI GESTIONE INTERRUPT
PROVENIENTI DA CLOCK E CONVERTITORE

```
                        *000 060
000 060 303   ADSVC,    JMP       SPAZIO NON SUFFICIENTE IN QUEST'AREA,
                                  COSICCHÈ
000 061 100             AHEAD     SI SALTA A UN INDIRIZZO SUPERIORE IN MODO
                                  DA
000 062 000             0         NON OCCUPARE L'INDIRIZZO 000 700, L'ALTRO
                                  INDIRIZZO DI VETTORE DI INTERRUPT.

                        *000 070
000 070 323   CLKSVC,   OUT       AZZERA IL FLAG DI INTERRUPT DAL CLOCK.
000 071 173             173
000 072 323             OUT       FA PARTIRE LA CONVERSIONE
000 073 037             037
000 074 373             EI        RIABILITA L'INTERRUPT
000 075 311             RET       RITORNA AL PROGRAMMA PRINCIPALE

                        *000 100
000 100 323   AHEAD,    OUT       AZZERA IL FLAG DI INTERRUPT
000 101 174             174       DAL CONVERTITORE
000 102 305             PUSHB     SALVA I REGISTRI
000 103 325             PUSHD
000 104 345             PUSHH
```

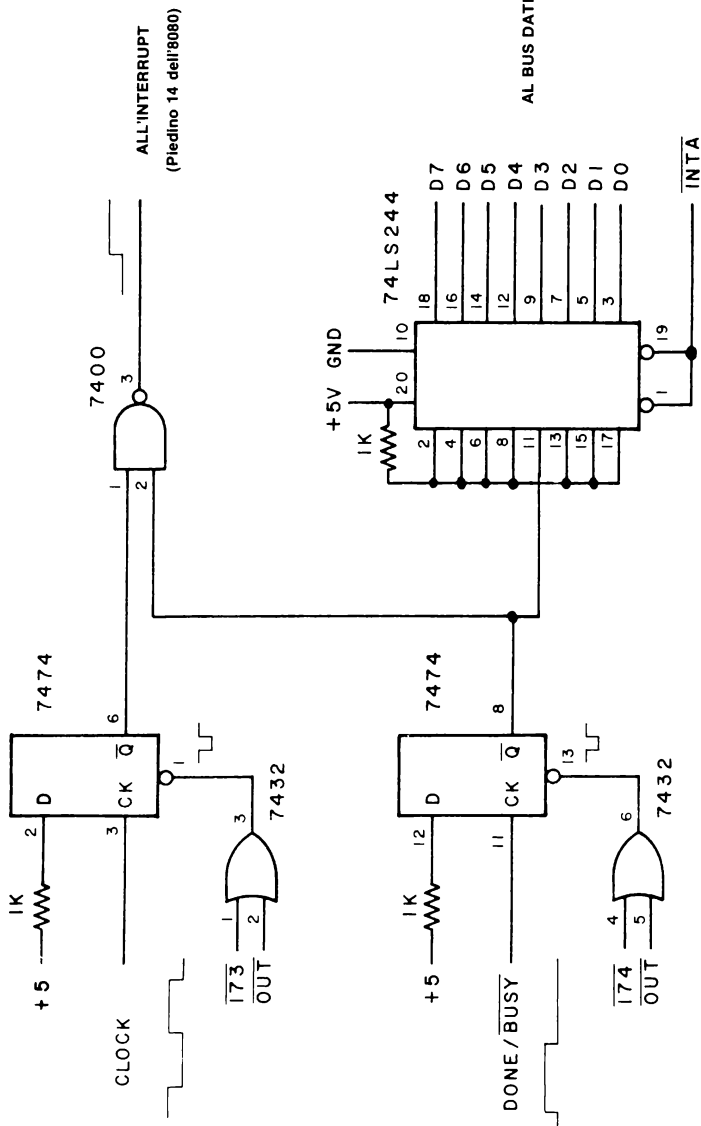



Figura 2-17. Schema di interrupt per due dispositivi, uno che genera l'istruzione RST 6 (367) e l'altro che genera l'istruzione RST 7 (377). Per un corretto funzionamento di questo circuito le richieste non possono essere simultanee.

```
000 105 365        PUSHPSW
000 106 345        PUSHH      SALVA H E L
000 107 365        PUSHPSW    SALVA A E I FLAG
000 110 315        CALL       SPOSTA I DATI IN GIU' DI UNA POSIZIONE
000 111 105        UPDATE     CORRISPONDE A DUE LOCAZIONI PER CREARE
                              SPAZIO
000 112 003        0          PER IL NUOVO DATO
000 113 052        LHLD       PRENDE IL PUNTATORE ALLA MEMORIA PER
000 114 056        NEWPNT     L'ACQUISIZIONE DI NUOVI DATI.
000 115 003        0
000 116 333        IN         LEGGE GLI 8 BIT MENO SIGNIFICATIVI
000 117 065        065
000 120 167        MOVMA      LI MEMORIZZA
000 121 043        INXH
000 122 333        IN         LEGGE I DUE BIT PIU' SIGNIFICATIVI
000 123 066        066
000 124 167        MOVMA      LI MEMORIZZA
000 125 361        POPPSW     RIPRISTINA I REGISTRI
000 126 341        POPH
000 127 321        POPD
000 130 301        POPB
000 131 373        EI         RIABILITA L'INTERRUPT.
000 132 311        RET
```

Se osservate attentamente i programmi riportati negli Esempi 2-14 e 2-15 noterete che è stato un buon uso delle istruzioni dell'8080 SHLD, LHLD e STA. Queste istruzioni ci hanno permesso di memorizzare gli indirizzi e le informazioni relative al contatore nella memoria a lettura/scrittura, in una serie di locazioni accessibili dai programmi. L'indirizzo dell'ultimo punto di lettura della temperatura, per esempio, è memorizzato nella locazione NEWPNT e a quella successiva. *Ricordatevi che un indirizzo a 16 bit richiede lo spazio di due byte di otto bit per la memorizzazione.* Quando si utilizza una tecnica di memorizzazione di indirizzi, come in questo caso, diventa semplice cambiare l'indirizzo di inizio del file di dati, memorizzato alla locazione POINT, e il numero della acquisizione, dal momento che il conteggio è memorizzato alla locazione COUNT. Dobbiamo tuttavia calcolare gli indirizzi delle ultime due locazioni del file che vengono utilizzate per memorizzare l'ultimo dato da acquisire. Ricordatevi che l'indirizzo dell'ultimo dato non deve variare una volta che si è determinato dal momento che quest'ultimo valore deve essere sempre posto nella stessa locazione del file, alle ultime due posizioni.

In molte applicazioni, la memorizzazione dei dati necessita di cambiamenti, cosicchè l'indirizzo di inizio, il numero di valori e gli indirizzi delle locazioni utilizzate per la memorizzazione dell'ultima lettura possono assumere dei valori diversi. Invece di calcolare continuamente questi indirizzi ogni volta che il programma viene utilizzato nelle diverse applicazioni, possiamo fare in modo che il calcolatore faccia tutto ciò per noi.

Nel prossimo esempio di programma, l'Esempio 2-16, l'unico START è stato cambiato così che il programma stesso calcola ora l'indirizzo dell'ultimo valore (due locazioni) una volta che sono stati acquisiti l'indirizzo di inizio del file e il numero di punti da rilevare dalle locazioni POINT (due byte) e COUNT (un byte). Questi valori possono essere immessi tramite un pannello di controllo, una telescrivente oppure possono essere caricati inizialmente dal programmatore. Dal momento che il resto del programma è lo stesso dell'Esempio 2-14, abbiamo riportato nell'Esempio 2-16 solo il cambiamento effettuato tra le locazioni START e CLEAR. Ricordatevi che se volete che l'indirizzo di inizio e il numero delle acquisizioni possa essere cambiato, l'area per la memorizzazione dei valori alle locazioni POINT, NEWPNT e COUNT deve essere in memoria a lettura/scrittura.

Alcuni dei passi di programma riportati nell'esempio 2-16 possono essere combinati come altri sottoprogrammi, ma abbiamo evitato ciò per non dover continuamente saltare avanti

e indietro tra vari segmenti del programma rendendo difficile la comprensione del programma. Noi comunque speriamo che riusciate ad utilizzare questi programmi, e a trovare altre soluzioni che possano essere più efficienti e applicabili alle vostre necessità specifiche.

```
                        ESEMPIO 2-16
                        PROGRAMMA PER INIZIALIZZARE INDIRIZZI E IL
                        CONTEGGIO
                        (PUO' ANCHE ESSERE USATO CON L'ESEMPIO 2-14)
                        SI ASSUME CHE VENGANO IMPIEGATI DUE BYTE
                        PER OGNI VALORE

                        *003 000
003 000 061   START,    LXISP     CARICA IL PUNTATORE DI STACK
003 001 370             370
003 002 004             004
003 003 373             EI        ABILITA L'INTERRUPT
003 004 052             LHLD      ORA, CALCOLA L'INDIRIZZO DELL'ULTIMO
                                  VALORE
003 005 060             COUNT     PRENDE IL VALORE ALLA LOCAZIONE COUNT
003 006 003             0
003 007 046             MVIH      PONE IL REGISTRO H A ZERO.
003 010 000             000
003 011 051             DADH      LO SOMMA A SE STESSO (MOLTIPLICA X 2)
003 012 353             XCHG      PONE IL VALORE RADDOPPIATO IN D,E
003 013 052             LHLD      PRENDE L'INDIRIZZO DI INIZIO DEL FILE
003 014 054             POINT
003 015 003             0
003 016 031             DADD      SOMMA IL NUMERO DEGLI INDIRIZZI.
003 017 053             DCXH      SOTTRAE 2
003 020 053             DCXH
003 021 042             SHLD      SALVA L'INDIRIZZO DELL'ULTIMO VALORE
003 022 056             NEWPNT    NELLA LOCAZIONE NWPNT E SUCCESSIVA
003 023 003             0
003 024 052             LHLD      PRENDE L'INDIRIZZO DEL FILE
003 025 054             POINT
003 026 003             0
003 027 072             LDA       PRENDE IL CONTATORE DI ACQUISIZIONI
003 030 060             COUNT
003 031 003             0
003 032 137             MOVEA
003 033 257   CLEAR,    XRAA      CONTINUA COME NELL'ESEMPIO PRECEDENTE

                        *003 054
003 054 000   POINT,    0         SE VOLETE EFFETTUARE CAMBIAMENTI, E
                                  PENSATE
003 055 000             0         DI UTILIZZARE QUESTO PROGRAMMA, QUESTE
                                  LOCAZIONI
003 056 000   NEWPNT,0            DEVONO ESSERE IN MEMORIA A
003 057 000             0         LETTURA/SCRITTURA.
003 060 000   COUNT,    0
```

L'esempio esaminato per l'acquisizione di valori di temperatura di una serra si suppone esegua l'acquisizione fino a un massimo di 256 valori. Ciò ha semplificato il programma in quanto vengono richieste operazioni matematiche di una singola precisione (cioè a un byte) per il conteggio dei dati.

Se si desidera lavorare con file più ampi i programmi devono essere modificati.

ANELLI CHIUSI - APPLICAZIONI DI CONTROLLI

L'esempio di misura della temperatura della serra mostra come un piccolo calcolatore può essere impiegato in una applicazione ad *anello aperto*, che non richiede cioè nessun tipo di controllo. I dati e i segnali di controllo hanno un'unica direzione, quella verso il calcolatore. Ci piacerebbe ora vedere se è possibile chiudere l'anello e permettere al calcolatore di eseguire qualche funzione di controllo per mantenere la temperatura all'interno di limiti prefissati.
Un ventilatore per il raffreddamento e un calorifero sono disponibili e possono essere controllati dal calcolatore.
Una interfaccia tipica di controllo è riportata nella Figura 2-18 dove un relè allo stato solido viene impiegato per il controllo del ventilatore, ma del tutto analoga può essere impiegata per il controllo del calorifero.
Il programma di controllo può essere scritto come sottoprogramma da aggiungere a quello di servizio dell'interrupt proveniente dal contatore, INTSVC, subito dopo che è stata chiamata la routine CONVRT. Così, l'operazione di controllo può essere effettuata basandosi sull'ultimo valore di temperatura acquisito. Il nostro sottoprogramma di controllo che abbiamo aggiunto lavora con dei valori arbitrari di temperature: 001 035 per il valore superiore e 002 310 per quello inferiore che corrispondono ai valori binari 0100011101 e 1011001000 rispettivamente. Con gli opportuni sensori si può far corrispondere questi valori alle temperature di 15° C (59° F) e 25° C (77° F).

ESEMPIO 2-17
SOTTOPROGRAMMA PER IL CONTROLLO DEL VENTILATORE E DEL CALORIFERO

```
                         *003 200
003 200 323   STPNT,     OUT         VENTILATORE SPENTO
003 201 317              317
003 202 323              OUT         CALORIFERO SPENTO
003 203 320              320
003 204 052              LHLD        PRENDE IL VALORE SUPERIORE A 10 BIT
003 205 251              HIVAL
003 206 003              0
003 207 174              MOVAH       QUESTE ISTRUZIONI REALIZZANO IL SUO
                                     VALORE
003 210 057              CMA         NEGATIVO
003 211 127              MOVDA
003 212 175              MOVAL
003 213 057              CMA
003 214 137              MOVEA
003 215 023              INXD
003 216 052              LHLD        PRENDE IL VALORE INFERIORE A 10 BIT
003 217 253              LOVAL
003 220 003              0
003 221 174              MOVAH       QUESTE ISTRUZIONI REALIZZANO IL SUO
                                     VALORE
003 222 057              CMA         NEGATIVO/ MENO UNO
033 223 107              MOVBA
003 224 175              MOVAL
033 225 057              CMA
003 226 117              MOVCA
003 227 052              LHLD        PRENDE IL NUOVO VALORE
003 230 255              NEWPNT
003 231 003              0
003 232 345              PUSHH       LO MEMORIZZA
```

```
003 233 031           DADD      SOMMA D,E con H,L
003 234 322           JNC       SE NON C'È IL CARRY, VALORE DI
003 235 243           TOLOW     TEMPERATURA TROPPO BASSA
003 236 003           0
003 237 323           OUT       C'È IL CARRY, QUINDI VALORE TROPPO ALTO
003 240 306           306       ACCENDERE IL VENTILATORE
003 241 341           POPH      RECUPERA IL REGISTRO H DALLO STACK IN
                                QUANTO NON SERVE
003 242 311           RET
003 243 341   TOLOW,  POPH      PRENDE NUOVAMENTE H PER ESEGUIRE IL
                                TEST.
003 244 011           DADB
003 245 330           RC        SE C'È IL CARRY, OK
003 246 323           OUT       NON C'È CARRY, TROPPO BASSO
003 247 307           307       ACCENDERE IL CALORIFERO
003 250 311           RET

003 251 310   HIVAL,  310       8 BIT MENO SIGNIFICATIVI DEL VALORE
                                SUPERIORE
003 252 002           002       2 BIT PIU' SIGNIFICATIVI DEL VALORE
                                SUPERIORE
003 253 035   LOVAL,  035       8 BIT MENO SIGNIFICATIVI DEL VALORE
                                INFERIORE
003 254 001           001       2 BIT MENO SIGNIFICATIVI DEL VALORE
                                INFERIORE.
003 255 000   NEWPNT,0
003 256 000           0
```

Il sottoprogramma di controllo, STPNT, realizza il confronto e l'accensione e lo spegnimento del calorifero o del ventilatore a seconda del caso. Ciascun dispositivo può essere acceso per un periodo prefissato oppure può essere acceso e lasciato acceso fino a che è stata raggiunta la temperatura richiesta. L'attuale metodo usato dipende sostanzialmente dalle dimensioni della serra; una piccola serra infatti può essere surriscaldata o raffreddata eccessivamente se si mantiene acceso uno dei due dispositivi fino alla lettura successiva che avviene ogni 10 minuti.
Il programma che abbiamo riportato nell'Esempio 2-17 ipotizza una serra grande per cui i dispositivi possono essere mantenuti accesi per l'intero periodo di campionamento di 10 minuti.

Mentre questo può sembrare un semplice programma, dal momento che la temperatura può essere controllata facilmente con un termostato, il programma di controllo e visualizzazione può diventare molto utile. Il valore di temperatura di ciascuna acquisizione può essere integrato per fornire il valore medio giornaliero, un'indicazione del calore prodotto dal sole per un dato giorno. Con un sistema che utilizza il riscaldamento e il raffreddamento di tipo solare, questo calcolatore può essere impiegato per il funzionamento del collettore solare o il controllo della sequenza della pompa, così come per l'acquisizione di dati. Una relazione del Copper Development Association indica l'impiego di un calcolatore programmabile Modello 31 della Tektronix proprio per un caso simile[1]

---

(1) CDA Decade 80 Solar House, Application Data Sheet, Copper Development Association, Inc., 405 Lexington Avenue, New York, NY 10017, 1977.

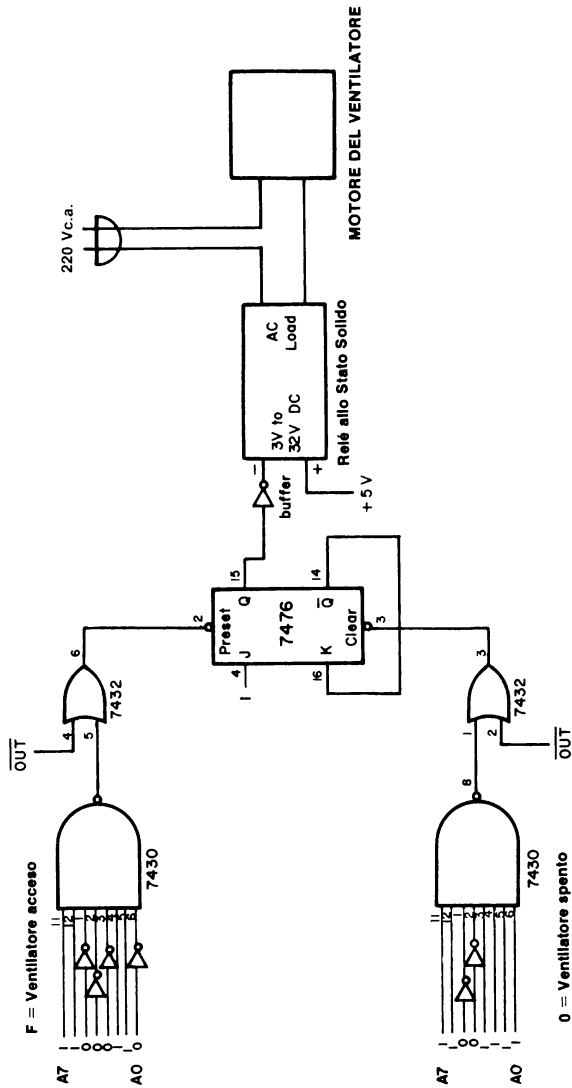



Figura 2-18. Impiego di un relé allo stato solido per il controllo di un ventilatore con un programma. Il comando OUT 306 accende il ventilatore mentre il comando OUT 317 lo spegne.

CAPITOLO 3

CONVERTITORI ANALOGICO DIGITALI A DOPPIA PENDENZA E PANNELLI DI MISURA DIGITALI

INTRODUZIONE

Il basso costo dei pannelli di misura digitali (DPM) li rende particolarmente interessanti per l'interfacciamento di segnali analogici con piccoli calcolatori. Essi forniscono una lettura diretta del valore misurato e molti modelli hanno uscite con valori BCD che facilitano ulteriormente l'interfacciamento. *La tecnica di conversione da analogico a digitale* con *integrazione di una doppia pendenza* è normalmente impiegata nel caso di pannelli di misura digitali. Convertitori A/D a doppia pendenza sono normalmente disponibili in moduli standard. La tecnica di conversione a doppia pendenza ha alcuni vantaggi rispetto a quella a rampa e quella ad approssimazioni successive che la rendono più interessante in alcune applicazioni.

CONVERTITORI ANALOGICO-DIGITALI A DOPPIA PENDENZA

La conversione a doppia pendenza sfrutta il principio di una misura indiretta di una tensione sconosciuta convertendole in un periodo di tempo. Uno schema a blocchi di un tipico convertitore di questo tipo è indicato in Figura 3-1.

Quando si deve realizzare una conversione, il convertitore a doppia pendenza collega tramite un commutatore elettronico la tensione sconosciuta con un circuito integratore. La logica di controllo permette all'integratore di integrare, o sommare, la tensione per un periodo di tempo fissato, T1, alla fine del quale l'ingresso dell'integratore è commutato dalla logica di controllo a una tensione di riferimento stabile con una polarità opposta di quella del segnale sconosciuto che deve essere misurato. Dal momento che l'ingresso dell'integratore è ora negativo, esso ha l'effetto di essere sottratto dalla somma accumulata durante il periodo di integrazione T1.

In quanto il potenziale di riferimento è stabile e conosciuto, la *pendenza* dall'uscita dell'integratore sarà costante nel tempo. Il convertitore a doppia pendenza misura l'intervallo di tempo T2, che è necessario per l'uscita dell'integratore per raggiungere il valore di zero volt. Queste relazioni temporali sono riportate nella Figura 3-2, dove è raffigurata la temporizzazione per due differenti valori di tensione A e B.

Quando viene applicato un valore di tensione più alto, A, all'integratore, il valore accumulato alla fine del periodo di integrazione, T1, è anch'esso maggiore. Dal momento che il potenziale di riferimento è sempre lo stesso, e di conseguenza la pendenza di "scarica" rimane uguale, il tempo necessario da parte dell'uscita dell'integratore di raggiungere lo zero viene aumentato fino al valore T3. Così una tensione, sconosciuta maggiore comporta un periodo di scarica proporzionalmente più lungo permettendo una misura indiretta del valore applicato al convertitore.

Durante il periodo di scarico, il segnale di clock è riportato tramite la logica di controllo ai contatori. Il periodo, T2, è misurato come numero di impulsi alla velocità per esempio,

di 100 impulsi per volt. Il conteggio accumulato così diventa una misura indiretta della tensione applicata al convertitore a doppia pendenza. Per un periodo di scarica più lungo, T3, la porta degli impulsi è aperta per un periodo superiore e vengono contati più impulsi in modo da indicare un valore di tensione da misurare superiore. La frequenza di clock e la logica di controllo vengono fissati in modo che la tensione applicata come ingresso al convertitore e gli impulsi del clock filtrati dalla logica di controllo siano proporzionali.

I contatori possono essere di tipo decimale, per un impiego di un pannello di misura digitale, oppure possono essere binari nel caso di applicazioni di controllo e di calcolatori dove non sia richiesta una codifica BCD.

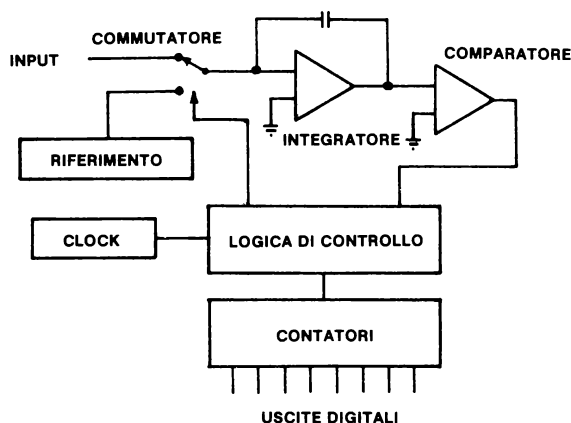



Figura 3-1. Schema a blocchi di un tipico convertiore analogico-digitale a doppia pendenza con uscita a otto bit.

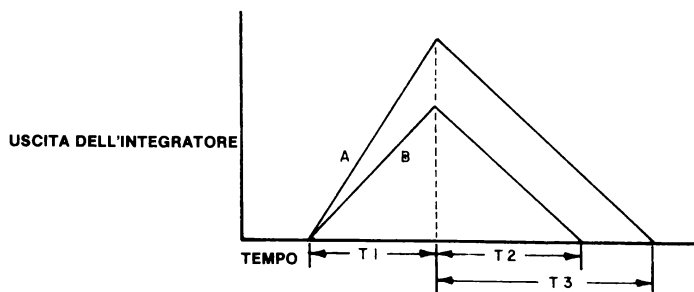



Figura 3-2. Diagramma dei tempi di un convertitore analogico-digitale a doppia pendenza. L'andamento si riferisce a due diversi valori di tensione A e B.

La tecnica descritta è analoga alla misura del flusso dell'acqua uscente da un tubo riempiendo un secchio in 60 secondi. Il volume di acqua raccolta è un'indicazione della entità del flusso, ma invece di misurarla direttamente, si effettua lo svuotamento ad una frequenza fissata, per esempio, un litro al secondo. Il flusso può essere allora indirettamente trovato dalla seguente formula:

$$\text{Flusso} = \frac{\text{tempo di svuotamento}}{\text{60 secondi}} \times 1 \text{ litro/secondo}$$

Nel caso di tensioni diventa:

$$E_{in} = \frac{T2}{T1} \times (V_{ref})$$

Il valore T1 è relativo al tempo di integrazione, mentre T2 a quello di scarica.

La maggior parte degli utilizzatori di convertitori troverà più comodo acquistare piuttosto che costruire dei dispositivi a doppia pendenza. Alcuni modelli rappresentati sono:

Analog Devices, Inc., Norwood, MA02602

ADC 1100	3 1/2 cifre 42 millisecondi di tempo di conversione campo di misura di ± 0,1999 volt
ADC 141	14 bit 40 millisecondi di tempo di conversione campo di misura di ± 10 volt

Analogic Corporation, Wakefield, MA 01880

AN 2313	10 bit 6,7 millisecondi di tempo di conversione
AN 2317	14 bit 6,7 millisecondi di tempo di conversione, ambedue i dispositivi hanno un campo di misura di ± 2 volt

Datel Systems, Inc., Canton, MA 02021.

ADC E10B	10 bit 1.25 millisecondi di tempo di conversione. Campo di misura di ± 1, ± 5 e ± 10 volt.

I convertitori A/D a doppia pendenza possono impiegare un lungo tempo per realizzare una sola conversione in quanto l'integrazione di riferimento e quella per la misura possono richiedere molto tempo. Alcuni dei modelli meno costosi a 10 e 12 bit hanno tempi di conversione che vanno da 5 a 10 millisecondi, mentre una risoluzione più alta può comportare tempi da 100 a 120 millisecondi. In contrasto con la loro relativa bassa velocità, i convertitori a doppia pendenza possono essere molto precisi.

Nell'intergrazione di un valore sconosciuto per uno specifico periodo di tempo, può succedere di integrare o sommare a zero disturbi periodici o casuali. In altre parole il disturbo viene conteggiato. Per esempio, consideriamo la misura del segnale, riportato nella Figura 3-3; al punto X. La tensione si suppone sia un valore continuo preciso e costante, ma sovrapposto c'è un disturbo ad una *frequenza di 60 Hz.*

Se un convertitore veloce a doppia pendenza inizia la sua conversione al punto X, il valore digitalizzato sarà troppo alto. La conversione non comporterà l'integrazione del disturbo se il periodo di integrazione, T1, è un multiplo intero del periodo di 60 Hz che è di 16,67 millisecondi.

Il disturbo a 60 Hz non influisce sul valore finale digitalizzato in quando la media del segnale di ingresso è quello che attualmente convertito, come riportato nella Figura 3-4.

Il valore ricavato dall'integrazione, cioè l'area sotto curva della Figura 3-4, è uguale all'area sotto la linea retta tratteggiata in quanto il disturbo a 60 Hz viene prima aggiunto e poi sottratto dall'accumulo totale. Nello stesso modo un disturbo casuale viene considerato, diminuendo la possibilità di un campionamento e di una conversione del segnale durante un disturbo impulsivo.

Tutto ciò viene chiamato "protezione ai disturbi in modo normale" *normal mode noise rejection*, che è definito come l'eliminazione di un disturbo sovrapposto a un segnale su un conduttore.

Molti convertitori A/D a doppia pendenza vengono normalmente predisposti per un'immunità a disturbi a 60 Hz a circa 40 dB, così come a 50 Hz dove tale frequenza è più comune.

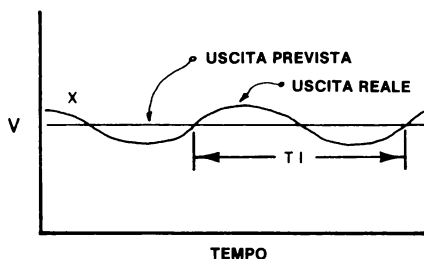



Figura 3-3. Diagramma di un segnale previsto e del segnale attuale che mostra il disturbo a 60 Hz sovrapposto.

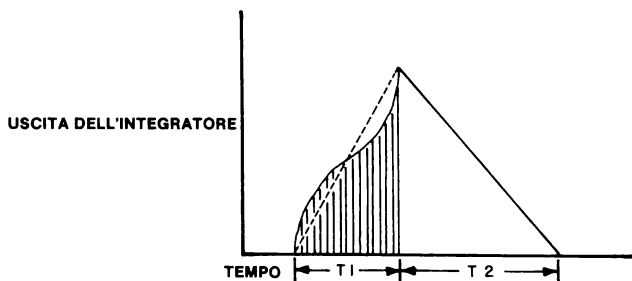



Figura 3-4. Integrazione di un segnale con sovrapposto un disturbo a 60 Hz. Il valore finale è lo stesso del segnale privo di disturbo. L'integrazione è realizzata in un periodo di tempo di 16,66 millisecondi.

Questi convertitori sono facilmente interfacciabili ai microcalcolatori impiegando la stessa tecnica che è stata sviluppata per l'interfacciamento del modulo di conversione ad approssimazioni successive. I segnali di controllo e i dati sono talmente simili che non è necessario affrontare nuovamente l'argomento. L'unica reale differenza tra i due argomenti è la tecnica di conversione adottata.

Il nostro principale interesse nel considerare i convertitori a doppia pendenza riguarda il loro impiego nei pannelli di misura digitali.

PANNELLI DI MISURA DIGITALI

I pannelli di misura digitali (DMP) sono convertitori analogico-digitali leggermente differenti da quelli discussi precedentemente, in quanto la visualizzazione è una parte integrante del convertitore. La maggior parte dei pannelli di misura digitali sono dispositivi che impiegano il codice B e D. L'uso di un DPM è frequentemente preferibile all'uso di un altro tipo di convertitore A/D, particolarmente in quelle applicazioni nelle quali la visualizzazione, l'indicazione decimale di una tensione, pressione, temperatura, o altre variabili si rende necessaria, in quanto una segnalazione di tipo numerico è formata da tutti i pannelli digitali.

Molti DPM vengono impiegati per sostituire misuratori analogici o a movimento magnetico. Tuttavia non necessariamente si prestano all'interfacciamento con il calcolatore, in quanto viene utilizzato il codice B C D e non sono disponibili uscite parallele; esistono comunque alcuni DPM che hanno caratteristiche che rendono possibile l'interfacciamento con sistemi a microprocessore. Le caratteristiche necessarie per effettuare l'interfacciamento sono:

- Uscite parallele in formato compatibile TTL
- Un segnale di stato per indicare la fine di una conversione A/D, anch'esso compatibile TTL.

La maggior parte dei pannelli di misura digitale, diversamente da altri convertitori A/D, sono dispositivi "free-running". Ciò significa che essi realizzano continuamente delle conversioni per aggiornare il visualizzatore. Queste conversioni sono controllate da un clock interno, ma molti DPM hanno una possibilità di conversione esterna che con un adeguato interfacciamento permette il comando da parte di altri dispositivi per l'esecuzione di una conversione. Dal momento che le uscite parallele del DPM sono in codice BCD, un rapido ripasso del sistema di codifica BCD viene riportato nel prossimo paragrafo. Nel caso siate già esperti nell'impiego di questa tecnica potete evitare di leggerlo in quanto si tratta di un rapido riassunto e non di un'esposizione dettagliata.

NUMERAZIONE DECIMALE CON CODIFICA BINARIA (BCD)

Nei sistemi decimali a codifica binaria, il raggruppamento decimale della cifra è mantenuto, ma ciascuna di esse è convertita nell'equivalente binario senza tenere conto della sua posizione all'interno del numero. In questo modo, un sette è sempre rappresentato da 0111_2, sia che sia il sette di 107 o di 7325. La possibilità di variazione della cifra è sempre limitata da zero a nove.

La numerazione BCD è facilmente ottenibile con l'aiuto della tabella riportata nel seguito. Dopo poche conversioni, sarete subito in grado di effettuare le successive senza l'aiuto della tabella.

Numerazione decimale con codifica binaria

Cifre decimali	Codice binario
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

Riportiamo due esempi di conversione da decimale a BCD:

Numero da convertire = 7325

0111 = 7   0011 = 3   0010 = 2   0101 = 5

Numero da convertire = 8039

1000 = 8   0000 = 0   0011 = 3   1001 = 9

E' importante da parte vostra tenere presente che poichè ciascuna cifra decimale, da zero a nove, può essere rappresentata da 4 bit, un microcalcolatore a 8 bit può memorizzare due cifre BCD in ciascuna posizione di memoria a 8 bit. Ciò significa un risparmio di metà della memoria nel caso in cui avessimo riservato una posizione per ogni cifra. Quando due cifre BCD sono memorizzate in questo modo si definiscono i dati *impaccati in codice BCD* e molto spesso si eseguono programmi per *impaccare* o, separare dati in codice BCD.

CONSIDERAZIONI SUI PANNELLI DI MISURA DIGITALI

La maggior parte dei pannelli di impiego generale sono dispositivi a 3 cifre e mezza o a 4 cifre e mezza. La cifra indirizzata come metà sta a indicare che ıa cifra più significativa può essere solo o uno o zero. Così, per un DPM a tre cifre e mezzo, il campo di variazione può essere da 0000 a 1999.
Alcuni DPM hanno un campo di variazione *unipolare*, cioè possono variare da zero al loro valore di fondo-scala, per esempio + 1,999 volt, mentre altri hanno un campo *bipolare*, cioè possono variare da un valore di fondo scala negativo e uno positivo, ad esempio ± 1,999 volt.

Altri pannelli possono essere programmati in modo da avere vari campi di variazione tramite un resistore o con connessioni digitali esterne. Alcuni dei più recenti sono *autoranging;* ciò significa che scelgono automaticamente il "range" corretto, e possono convertire segnali analogici in un campo di ± 19,99 volt, ± 1,999 volt e ± 199,9 millivolt. I DPM "autoranging" sono leggermente più costosi che i moduli a range fisso, ma possono essere particolarmente utili in applicazioni dove la variazione dei segnali può essere di alcune decadi.

I pannelli di misura digitali sono generalmente realizzati in modo da avere tempi di conversione lunghi se confrontati con i veloci convertitori ad approssimazioni successive o a rampa. La maggior parte impiega parecchi millisecondi per realizzare una conversione e la velocità di ripetizione di una misura varia da 2 a 20 conversioni al secondo. Per alcuni sistemi ciò può essere considerata una velocità lenta, ma in altri in cui non sono

necessari tempi di conversione molto elevati, i DPM vengono usati su larga scala. Nell'esempio considerato riguardante le misure delle temperature di una serra, sarebbe potuto essere usato, con lo stesso successo di un convertitore ad approssimazioni successive, a 10 bit, un DPM a 3 cifre e mezzo. Nel caso della serra le temperature sono state registrate ogni 10 minuti ed è improbabile che la temperatura subisca cambiamenti veloci durante il tempo di conversione necessario ad un DPM che è dell'ordine di parecchi millisecondi.

Possiamo riassumere il contenuto di queste pagine elencando le più importanti caratteristiche dei pannelli digitali:

- Uscite BCD parallele TTL - compatibili (opzionale su alcuni)
- Disponibile in diversi campi di variazione dei quali sono programmabili
- Segnalazione del flag di fine-conversione tramite flag (EOC)
- Ingresso di inizio conversione esterna (opzionale su alcune)
- Lento da due a 20 conversioni al secondo
- Visualizzazione di dati in volt o altra unità di misura
- Convertitore analogico-digitale a basso costo con condizionamento di segnale analogico incorporato.

INTERFACCIAMENTO DEL PANNELLO DI MISURA

I pannelli digitali vengono interfacciati con i calcolatori in un modo molto simile a quello usato nei precedenti esempi di interfacciamento di un convertitore analogico-digitale. Le uscite del DPM sono disponibili in modo parallelo, compatibile TTL; il DPM fornisce un segnale in uscita di fine-conversione per indicare che le uscite digitali parallele sono valide e che possono essere campionate dal calcolatore. In base al fatto che il DPM sia interfacciato o no, è la logica del DPM interna o il calcolatore che fornisce l'impulso per dare il via ad una conversione.

In alcuni DPM ci sono, comunque, altri segnali che devono essere ugualmente utilizzati. Alcuni di loro devono essere collegati sia ad un uno che ad uno zero logico in modo da selezionare un campo di variazione, programmare un punto decimale sulla visualizzazione, ecc. Questi ingressi non richiedono altri interventi ed è improbabile che possano essere cambiati una volta che sono stati collegati al sistema.

Esistono altri segnali molto importanti, messi in uscita dal DPM per indicare campo, supero di capacità, polarità, ecc., che non si trovano generalmente nei piccoli convertitori modulari per cui non ne abbiamo parlato in nessuno dei precedenti esempi di interfacciamento di un convertitore analogico-digitale.

Vogliamo invece illustrare qui due esempi di interfacciamento di un pannello digitale: 1) un'interfaccia semplice di un DPM standard con un campo di misura fisso e 2) un esempio più complesso nel quale è stato impiegato un DPM (con scelta automatica del campo di misura). Questi tipi di interfacciamento non sono limitati ai soli DPM, dato che allo stesso modo possono essere interfacciati anche altre fonti di dati BCD. Misuratori di frequenza, clock, contatori di eventi e altri dispositivi, compresi gli strumenti analitici come spettometri e cromatografi, fanno parte di questa categoria.

INTERFACCIAMENTO DI UN PANNELLO DIGITALE - ESEMPIO N. 1

Nell'esempio che segue, abbiamo interfacciato un pannello con un singolo campo bipolare di ± 1,999 volt con un microcalcolatore basato sull'8080. Il DPM usato in questo caso è un modello da 3 cifre e mezzo, AD2009 della Analog Devices Inc.

Il software necessario per acquisire i dati provenienti dal modello AD2009 viene scritto sotto forma di sottoprogramma generale che può essere usato anche da altri programmi. L'hardware è realizzato in modo che ogni cifra BCD ha la propria porta di ingresso three-state. L'AD2009 può essere fatto funzionare sia con sincronizzazione esterna che interna.

Abbiamo scelto di usare il sincronismo interno, che esegue sei conversioni al secondo (o una conversione ogni 167 millisecondi). Il sincronismo esterno può essere utile quando è necessario che il DPM esegua una conversione su comando del calcolatore.

Con l'aiuto di un sincronismo interno, il programma sarà in grado di ammettere dati BCD sia della conversione che ha appena avuto luogo che della conversione in corso. Così, se succede che il calcolatore richieda dati dal DPM immediatamente dopo il completamento di una conversione, i dati sono disponibili e immessi nel calcolatore. Se, tuttavia, il calcolatore richiede dati mentre il blocco conversione del DPM sta elaborando il segnale analogico, il computer deve attendere fino alla fine della conversione in corso prima che i dati siano utilizzabili.
Ciò fa sì che nell'acquisizione dei dati ci sia un'incertezza massima di 167 millisecondi, valore accettabile quando si ha a che fare con applicazioni molto lente.

INTERFACCIAMENTO DELL'AD2009 Il pannello digitale AD2009 ha un numero di linee di ingresso e uscita compatibile con i livelli TTL standard presenti nel calcolatore basato sull'8080 che stiamo usando. Alcuni segnali del DPM non hanno bisogno di essere controllati dal calcolatore per cui possono essere collegati sia allo zero che all'uno logici a seconda del controllo necessario.

Qui di seguito sono elencati i segnali di interfacciamento più importanti dell'AD2009:

Output digitali

Dati BCD		12 linee
OVERRANGE	(cifra più significativa) 1 logico = 0*nnn* volt 0 logico = 1*nnn* volt *n*=da 0 a 9	1 linea
Superamento di capacità		
(OVERLOAD)	1 logico = All'interno del campo di misura 0 logico = All'esterno del campo di misura	1 linea
POLARITA'	1 logico = Valore di tensione di ingresso positiva 0 logico = Valore di tensione di ingresso negativa	1 linea
STATO	1 logico = Conversione in corso 0 logico = Conversione eseguita	1 linea

Input Digitali

HOLD	1 logico = Modo di conversione normale 0 logico = Mantiene i dati, disattiva il trigger	1 linea

L'input HOLD è molto utile in questa applicazione, poichè permette al calcolatore di disattivare il processo di conversione in modo che i dati possano essere letti sotto controllo di programma. Quando l'input HOLD del DPM viene portato allo zero logico, non ci possono essere altre conversioni e i dati in uscita vengono mantenuti stabili o memorizzati in modo da poter essere letti.

L'interfaccia di un AD2009 è stata realizzata usando quattro porte di un ingresso three-state a quattro bit. Le tre cifre BCD vengono immesse una alla volta sotto controllo software.
I quattro segnali di stato, OVERRANGE, OVERFLOW, POLARITA' e STATUS vengono immessi tramite una porta di ingresso three-state a quattro bit separata. Nonostante l'8080

sia un calcolatore a otto bit, può gestire facilmente byte a quattro bit, quindi più piccoli. Anche se può apparire poco efficace interfacciare il DPM in questo modo è molto facile farlo e costa all'incirca quanto costa per due porte di ingresso a 8 bit.
Avrete notato che nell'interfaccia di controllo dello stato dell'ingresso HOLD del DPM è compreso un flip-flop. L'istruzione OUT 061 forza il DPM nello staio HOLD mentre l'istruzione OUT 060 toglie il segnale HOLD e permette al DPM di funzionare normalmente. Nella Figura 3-5 è illustrato uno schema completo dell'interfaccia del DPM. La disposizione dei segnali sui piedini di uno solo degli 8095 buffer three-state è però contenuta nel disegno, gli altri dispositivi hanno le configurazioni dei piedini esattamente corrispondenti.

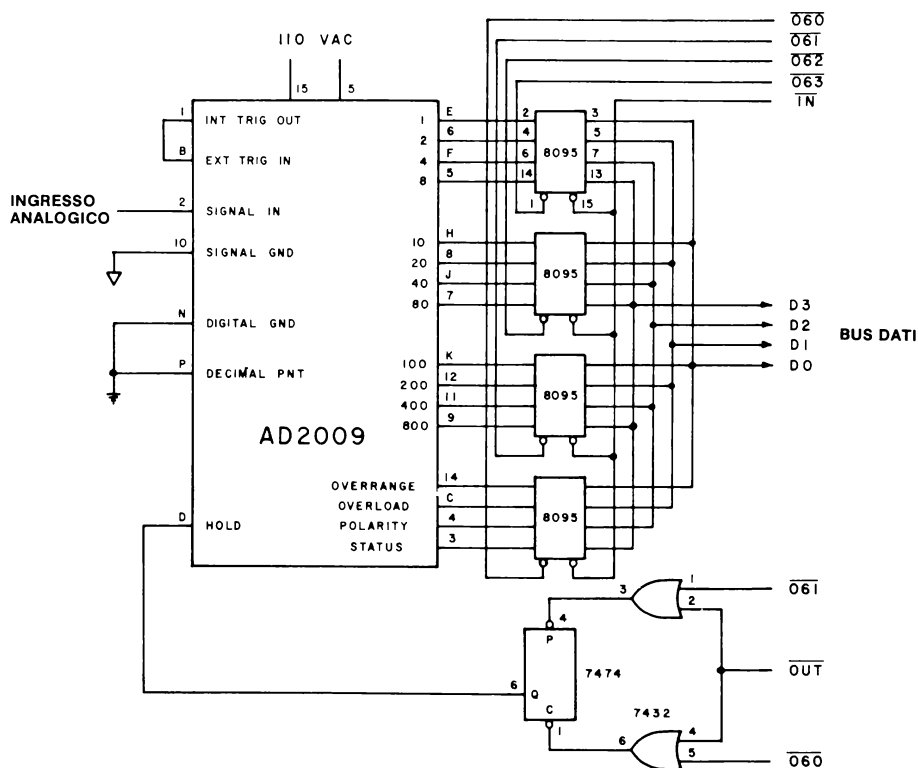



Figura 3-5. Interfaccia completa del pannello digitale AD2009 della Analog Devices Inc.

PROGRAMMA PER L'AD2009 Il programma di controllo dell'interfaccia del DPM AD2009 è forse la parte più importante dello schema completo. Il sottoprogramma di controllo dell'interfaccia di un DPM per un impiego generale deve avere i seguenti requisiti:

1. Immettere i dati provenienti dal DPM
2. Memorizzare i dati in cinque successive locazioni di memoria.

Locazione	Dati
n.	
n + 1	Bit di polarità (segno)
n + 2	Bit di overrange
n + 3	100 (di 100)
n + 4	10
	1

3. Se si ha una condizione di superamento di capacità uscire da sottoprogramma per ritornare al programma principale con il bit del CARRY posto a 1, altrimenti ritornare con i dati memorizzati in una lettura/scrittura e il bit del CARRY uguale a zero.

Il bit di polarità e il bit di "overrange" verrano memorizzati nella posizione D0 del bit nelle loro rispettive locazioni di memoria. Gli altri sette bit in ciascuna locazione invece non verrano usati. Va ricordato che il bit del carry è un flag interno dell'8080, abbinato, di solito, all'accumulatore o registro A.

Sarà compito del programma principale quello di richiamare il sottoprogramma del DPM per controllare il flag del carry per la condizione di superamento di capacità.
Poichè la virgola decimale è collocata tra le centinaia e le migliaia, non occorre memorizzare un carattere per rappresentarla.

Il sottoprogramma del DPM dovrebbe essere modificato per localizzare e agire su una condizione di superamento di capacità se il sottoprogramma è usato sotto controllo di interrupt.
Poichè gli interrupt possono verificarsi in qualunque momento, non possiamo essere sicuri che il programma che è in corso di esecuzione sarà in grado di agire su una condizione di superamento di capacità. Va ricordato, inoltre, che se il flag del carry è cambiato durante il sottoprogramma e il sottoprogramma di servizio dell'interrupt ritorna al programma principale con il carry, o altri registri nello stato errato, l'effetto sul programma può essere disastroso. La maggior parte delle applicazioni con il DPM saranno abbastanza lente da non richiedere l'uso dell'interrupt.

Il sottoprogramma completo di un DPM è illustrato nell'Esempio 3-1. Il sottoprogramma controlla la linea di STATO per stabilire se il DPM ha completato la conversione in corso. Una volta che è stata indirizzata la condizione di fine-conversione, viene attivata la linea HOLD del DPM e il calcolatore immette i dati BCD. Il programma di quest'ultimo per prima cosa controlla l'ingresso di superamento di capacità dato che una condizione di questo genere significa semplicemente il porre il flag del carry a uno logico e il ritorno al programma principale. Solo se non viene riscontrata nessuna condizione di superamento il calcolatore passerà all'acquisizione dei dati e ai passi di memorizzazione.

ESEMPIO 3-1
SEMPLICE SOTTOPROGRAMMA PER IL CONTROLLO DI UN DPM

```
                        * 030 000
030 000 345    DPM,     PUSHH       SALVA I REGISTRI NELLO STACK
```

```
030 001 305            PUSHB
030 002 041            LXIH     PUNTA ALL'AREA DI MEMORIA
030 003 000            STORE
030 004 040            0
030 005 016            MVIC     ATTIVA UNA MASCHERA PER I 4 BIT MENO
                                SIGNIFICATIVI
030 006 017            017      MASCHERA=00001111
030 007 333    TEST,   IN       ACQUISISCE E CONTROLLA IL BIT DI STATO
030 010 060            060
030 011 346            ANI      CONTROLLA LO STATO
030 012 010            010
030 013 302            JNZ      SE NON È STATO FATTO, LO CONTROLLA DI
                                NUOVO
030 014 007            TEST
030 015 030            0
030 016 323            OUT      ESEGUITO, METTE IL DPM NEL MODO HOLD
030 017 060            060
030 020 333            IN       ACQUISISCE LO STATO AGGIORNATO
030 021 060            060
030 022 107            MOVBA    LO MEMORIZZA
030 023 346            ANI      CONTROLLA SE ESISTE UN SUPERAMENTO DI
                                CAPACITA'
030 024 002            002
030 025 302            JNZ      SE NON C'È, VA ALL'OK
030 026 034            OK,
030 027 030            0
030 030 301            POPB     SE C'È RIPRISTINA I REGISTRI
030 031 341            POPH
030 032 067            STC      ATTIVA IL CARRY E
030 033 311            RET      RITORNA
030 034 170    OK,     MOVAB    PRENDE LA PAROLA STATUS
030 035 037            RAR      FA RUOTARE IL BIT DELLA POLARITA' ALLA
                                POSIZIONE D0
030 036 037            RAR
030 037 346            ANI      MASCHERA TUTTI GLI ALTRI BIT TRANNE D0
030 040 001            001
030 041 167            MOVMA    LO MEMORIZZA
030 042 043            INXH     INCREMENTA IL PUNTATORE ALLA MEMORIA
030 043 170            MOVAB    PRENDE DI NUOVO LO STATUS
030 044 346            ANI      MASCHERA TUTTO ALL'INFUORI DEL BIT DI
                                OVERRANGE
030 045 001            001
030 046 167            MOVMA    LO MEMORIZZA
030 047 043            INXH
030 050 333            IN       ACQUISISCE LA CIFRA SUCCESSIVA
030 051 061            061
030 052 241            ANAC     MASCHERA I BIT NON DESIDERATI
030 053 167            MOVMA    MEMORIZZA IL RISULTATO
030 054 043            INXH
030 055 333            IN       ESEGUE PER LA CIFRA SUCCESSIVA
030 056 062            062
030 057 241            ANAC
030 060 167            MOVMA
030 061 043            INXH
030 062 333            IN       ESEGUE PER L'ULTIMA CIFRA
030 063 063            063
030 064 241            ANAC
030 065 167            MOVMA
030 066 323            OUT      TOGLIE IL MODO HOLD
```

```
030 067 061              061
030 070 067              STC       TOGLIE IL CARRY PONENDOLO A UNO
030 071 077              CMC       E COMPLEMENTANDOLO
030 072 301              POPB      RIPRISTINA I REGISTRI
030 073 341              POPH
030 074 311              RET

                         AREA DI MEMORIZZAZIONE DEI DATI DI DPM. DEVE ESSERE
                         IN UN'AREA DI MEMORIA A LETTURA/SCRITTURA

                         * 040 000
040 000 000     STORE,   0         IL SEGNO È MEMORIZZATO QUI
040 001 000              0         BIT DI OVERRANGE
040 002 000              0
040 003 000              0
040 004 000              0
```

INTERFACCIAMENTO DI UN PANNELLO DIGITALE - ESEMPIO N. 2

In questo esempio abbiamo deciso di utilizzare un pannello con scelta automatica del campo di variazione tra tre campi di misura di ± 19,99 volt, ± 1,999 volt e ± 0,1999 volt (± 199,9 millivolt), e abbiamo scelto il modello DM2000AR della Datel Systems, Inc., che meglio si adatta allo scopo.

Gli ingressi e uscite digitali che seguono sono forniti per l'interfacciamento, ma anche per altri usi.

Output digitali

Dati BCD (tre cifre)		12 linee
Dati da mezza cifra		1 linea
POLARITA'	1 logico=Valore di tensione di ingresso positiva 0 logico=Valore di tensione di ingresso negativa	1 linea
OVERFLOW	1 logico=All'interno del campo di misura 0 logico=All'esterno del campo di misura	1 linea
DONE	1 logico=Conversione in corso 0 logico=Conversione effettuata	1 linea
PUNTO DECIM.		3 linee
	Campo = 19,99 volt, 0 logico al piedino A6 Campo = 1,999 volt, 0 logico al piedino A5 Campo = 0,1999 volt, 0 logico al piedino A7	

Nota: Un solo segnale per volta è posto a zero

Input digitali

MODO RANGE	1 logico=Scelta del campo esterno 0 logico=Scelta del campo automatica	1 linea
R1 e R2, controllo del campo esterno		2 linee
	Range = 19,99 volt, R1=0, R0=1 Range = 1,999 volt, R1=1, R0=0 Range = 0,1999 volt, R1=0, R0=0	

Nota: R1=R0=1 non è ammesso

START — Una transizione positiva riattiva il DPM — 1 linea
Una transizione negativa dà il via ad una conversione
Minima ampiezza di impulso = 100 nanosecondi

START INTERNO
1 logico=Esecuzione — 1 linea
0 logico=Stop

Ci sono pochi altri segnali del DM2000AR che però non vengono usati in questa configurazione di interfaccia. L'interfaccia di un microcalcolatore deve essere in grado di attivare una conversione, "sentire" che la conversione è stata completata e immettere i dati. In questo caso, esistono 13 linee di dati numerici per tre cifre e mezzo, una linea di superamento di capacità per la polarità e tre linee per la posizione della virgola decimale. Anche il flag DONE del DM2000AR deve essere immessa.

Qui abbiamo deciso di collegare i segnali del MODO RANGE, R1 e R0, e terza, 0 allo zero logico, e il segnale INTERNAL START a + 5 volt, o 1 logico per configurare il DPM nel modo automatico.
Alcuni di questi ingressi digitali possono essere controllati dal calcolatore, se lo si desidera, usando una porta di uscita con memorizzazione per fornire i segnali logici necessari.

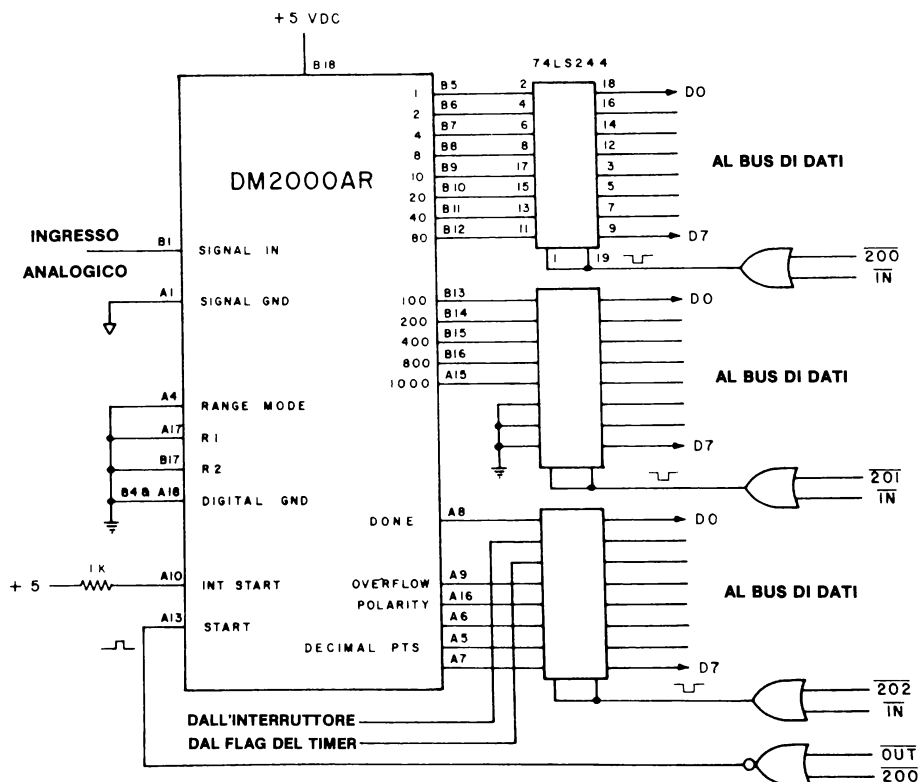



Figura 3-6. Interfaccia completa del pannello di misura DM2000AR della Datel System

In quest'applicazione specifica, non è stato necessario. La conversione del DPM viene iniziata con un impulso OUT200 fornito eseguendo l'operazione con l'indirizzo 200 che non corrisponde a nessun dispositivo.

Abbiamo deciso di usare, in questo esempio, la procedura di I/O tramite accumulatore, nonostante che l'I/O in mappa di memoria è ugualmente valido in un'applicazione come questa.
Nella Figura 3-6 è illustrato un tipico esempio di interfaccia per il DM2000AR che utilizza tre porte di ingresso a otto bit realizzate con buffer three-state SN74LS244 della Texas Instruments. Poichè si tratta di un circuito integrato relativamente nuovo, diamo nella Figura 3-7 una rappresentazione schematica.

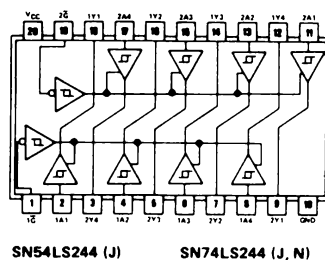



Figura 3-7. Rappresentazione schematica di un circuito integrato con buffer three-state SN74LS244. Per gentile concessione della Texas Instruments Incorporated di Dallas, TX.

Alle tre porte di ingresso sono stati assegnati gli indirizzi 200, 201 e 202. Dovreste ora essere in grado di indirizzare delle somiglianze dirette tra questo circuito, l'interfaccia per il convertitore A/D a 10 bit presentato nella prima parte e l'interfaccia per il DPM appena presentata. La stessa somiglianza è riscontrabile anche a livello di programmi.

PROGRAMMI PER IL DPM DM2000AR

Il programma usato per il controllo del pannello di misura DM2000AR può essere molto semplice ma anche molto complesso, a seconda del tipo di lavoro che occorre eseguire. Ancora una volta i tre compiti basilari sono quelli di:

- Iniziare una conversione
- Individuare il flag di fine-conversione (eseguita)
- Se eseguita, immettere i dati.

La differenza maggiore tra il DM2000AR e il pannello di misura AD2009 è che il DM2000AR hanno uscite addizionali per la posizione della virgola decimale, un'indicazione del campo di tensione attualmente in uso per il DPM.
Poichè per questa applicazione è stato scelto il DM2000AR, dobbiamo dare per scontato che quest'informazione è utile.

In questo esempio noi useremo effettivamente il DPM, il microcalcolatore e l'interfaccia per costruire un registratore di dati, ossia uno di quei dispositivi o sistemi che memorizzano dati in una forma utile in un secondo tempo, quando il dato viene valutato effettivamente. I registratori di dati non prendono di solito delle decisioni basate sui dati che stanno raccogliendo, per cui non hanno bisogno di un calcolatore molto sofisticato per l'eleborazione dei dati, di conseguenza il microcalcolatore può essere adatto a questo scopo.

I questo caso, stamperemo i dati che vengono raccolti su una telescrivente per impiegarli successivamente. Poichè la maggior parte delle telescriventi del tipo ASR perforano i caratteri su nastro mentre li stampano, è possibile una registrazione dell'informazione compatibile con il calcolatore, permanente e completa.

Dopo aver interfacciato il microcalcolatore, come risulta nella Figura 3-6, è ora necessario sviluppare il programma necessario alla registrazione delle funzioni. A questo proposito occorre tener conto delle seguenti necessità:

- Far partire la registrazione dei dati solo dopo che l'interruttore è stato chiuso.
- Campionare i dati ad una velocità regolata da un clock esterno. La velocità campione andrà a due acquisizioni al secondo programma al minuto.
- Stampare i dati nel formato +*nnnn*, con la virgola decimale nell'esatta posizione adatta al campo di tensione che si sta misurando.
- Stampare cinque colonne di dati traversalmente rispetto alla pagina della telescrivente, con tre spazi tra una colonna e l'altra.
- Stampare "OVER" quando si presenta la condizione di superamento di capacità.

Si presuppone che i nostri lettori siano in grado di procurarsi il circuito di clock e la chiusura ad interruttore a zero o a terra. (Questi concetti sono stati già esaminati nella seconda parte). La chiusura ad interruttore è stata usata anche per eseguire il reset del circuito di clock in modo da sincronizzare il clock con il computer. In questo modo saremo sicuri che il primo periodo di clock inizia da $t = 0$. La velocità di clock viene determinata collegando il flag del clock all'uscita opportuna della rete del divisore/oscillatore, come già indicato nella Figura 2-10 e 2-11. Due delle posizioni del bit inutilizzate nella porta di ingresso 202 vengono usate per controllare la chiusura dell'interruttore e il flag del clock.
La posizione D1 è assegnata all'interruttore, mentre la posizione D2 del bit è assegnata al flag del clock. Poiché nel circuito del timer viene usato il flag del flip-flop, nell'interfaccia deve esserci un impulso di azzeramento del flag. In questo caso abbiamo scelto di far eseguire all'impulso OUT 200 due funzioni: far partire una conversione del DPM DM2000AR e azzerare i flip-flop del flag del clock.

La circuiteria di clock e di flag non costituisce oggetto di particolare interesse di questo volume: la nostra attenzione si è concentrata piuttosto sul programma necessario ad eseguire le funzioni di registrazione dei dati. A questo proposito nella Figura 3-8 troverete un esempio semplificato di flow-chart. Di fatto il programma è in un certo senso più complesso dato che i dati BCD devono essere convertiti nei codici ASCII corrispondenti a ogni cifra decimale, e il segno deve essere posto in uscita come + o − e il punto decimale deve essere posto in modo corretto tra le cifre. Deve inoltre essere individuata la condizione di superamento delle capacità su cui si deve agire.

La conversione di dati BCD nel corrispondente codice ASCII (American Standard Code for Information Interchange) è lineare. Il codice ASCII, più usato dalla maggior parte delle telescriventi e dei terminali, è illustrato, assieme al codice binario a otto bit e agli equivalenti ottali per i caratteri decimali da zero a nove, nella Figura 3-9. Dovrebbe essere facile notare che gli ultimi quattro bit in ciascun codice ASCII sono l'equivalente BCD della cifra decimale rappresentata. La conversione ha luogo semplicemente aggiungendo il valore 10110000_2, o 260_8 al valore BCD.

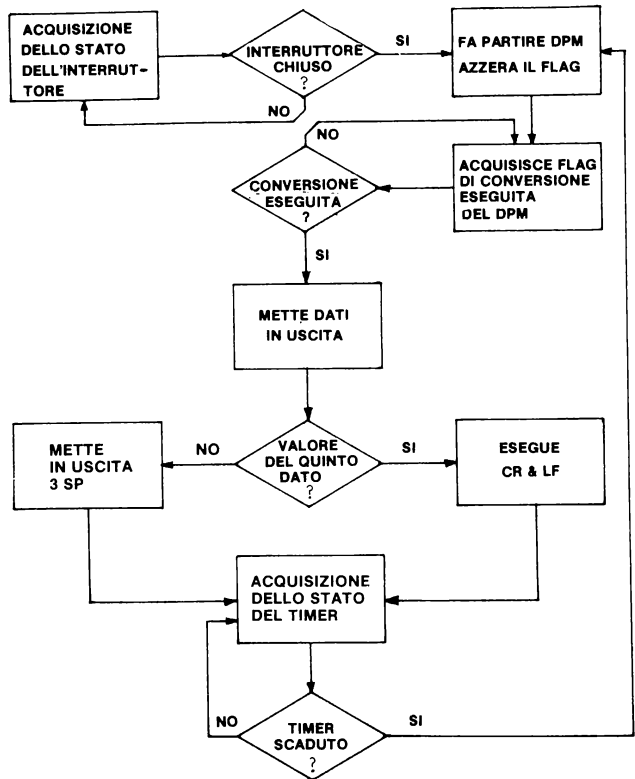



Figura 3-8. Flow-chart per il programma di controllo del registratore di dati.

Decimale	BCD	ASCII	
		Binario	Ottale
0	0000	10110000	260
1	0001	10110001	261
2	0010	10110010	262
3	0011	10110011	263
4	0100	10110100	264
5	0101	10110101	265
6	0110	10110110	266
7	0111	10110111	267
8	1000	10111000	270
9	1001	10111001	271

Figura 3-9. Tabella del Codice ASCII, che illustra il rapporto esistente tra i caratteri ASCII e i loro equivalenti ottali delle cifre decimali da zero a nove. Notate che abbiamo ammesso che il bit D7 = 1. Ciò può essere vero in alcuni sistemi che usano un bit di parità.

Il programma destinato all'uscita su telescrivente e alla formattazione dei dati é più complesso della parte del programma relativa all'acquisizione dei dati, come risulta dalla forma del flow-chart nella Figura 3-10. Il listing completo del programma per l'acquisizione e il programma di uscita usato per registratore dati é illustrato nell'Esempio 3-2.

Parte del programma usato nell'esempio del registratore di dati può essere interessante dato che può essere usato anche per altri problemi simili. Nei prossimi capitoli vedremo in dettaglio tre delle parti del programma.

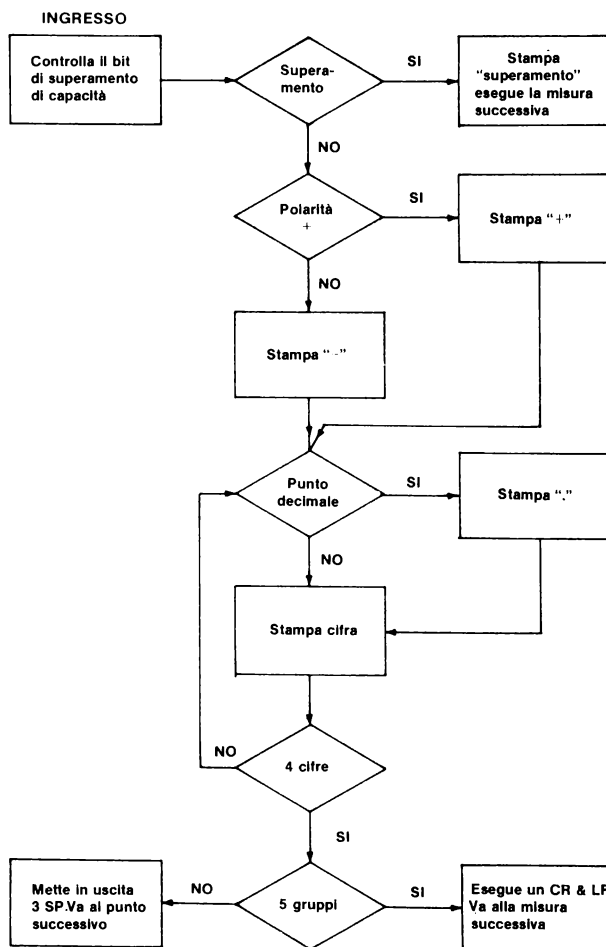



Figura 3-10. Flow-chart per la parte di controllo della telescrivente del programma del registratore di dati.

GESTIONE DEI DATI BCD

I dati in codice BCD richiedono solo 4 bit per ogni cifra, così piuttosto che sprecare una parola di otto bit per altri 4 bit, i dati vengono impaccati in modo da riservare una sola locazione di otto bit per due valori. Ciò viene realizzato realizzando l'interfaccia in modo da acquisire due cifre BCD tramite una porta in ingresso a 8 bit. In ogni caso, la cifra più significativa (MSD) delle due é rappresentata dai bit delle posizioni che vanno da D4 a D7. Per convertire questi 4 bit in un carattere ASCII che rappresenti la cifra corrispondente, i bit D4-D7 devono essere ruotati a destra nelle posizioni D3-D0. Questo nuovo valore é successivamente mascherato per eliminare i 4 bit D7-D4, tramite l'istruzione di AND logico con il valore di maschera 00001111_2. L'aggiunta del valore 10110000_2 ai quattro bit di dato, attualmente nelle posizioni D0-D3, converte la cifra BCD nel codice ASCII corrispondente.

```
                          ESEMPIO 3-2
                          IMPIEGO DEL MICROCALCOLATORE COME REGISTRATORE
                          DI DATI

                          *030 000
030 000 061               LXISP
030 001 377               377
030 002 030               030
030 003 315               CALL      INIZIA CON UNA LINEA NUOVA
030 004 163               CRLF
030 005 030               0
030 006 333     TEST,     IN        ESEGUE IL TEST PER LA CHIUSURA
                                    DELL'INTERRUTTORE
030 007 202               202
030 010 346               ANI
030 011 002               002
030 012 302               JNZ
030 013 006               TEST
030 014 030               0
030 015 056     INIT,     MVIL      INIZIALIZZA IL CONTATORE DI COLONNE
030 016 005               005
030 017 333     TIMER,    IN        ESEGUE IL TEST SUL FLAG DEL TIMER
030 020 202               202
030 021 346               ANI
030 022 004               004
030 023 312               JZ
030 024 017               TIMER
030 025 030               0
030 026 323               OUT       AZZERA IL FLAG DEL TIMER E ATTIVA IL DPM
030 027 200               200
030 030 333     FLAG,     IN        ESEGUE IL TEST SUL FLAG EOC
030 031 202               202
030 032 107               MOVBA     MEMORIZZA I BIT DI STATO
030 033 017               RRC
030 034 332               JC
030 035 030               FLAG
030 036 030               0
030 037 333               IN        CONVERSIONE COMPLETATA, IMMETTE
030 040 200               200       LA PRIMA PAROLA DI OTTO BIT
030 041 137               MOVEA     LA MEMORIZZA NEL REGISTRO Z
030 042 333               IN        IMMETTE LA SECONDA PAROLA DI OTTO BIT
030 043 201               201
030 044 127               MOVDA     LA MEMORIZZA NEL REGISTRO D
030 045 170               MOVAB     ACQUISISCE I BIT DI STATO
```

```
030 046 346             ANI       ESEGUE IL TEST SUPERAMENTO DI CAPACITA'
030 047 010             010
030 050 312             JZ        SE ESISTE, SALTA
030 051 220             OVER
030 052 030             0
030 053 170             MOVAB     ACQUISISCE DI NUOVO I BIT DI STATO
030 054 016             MVIC      ATTIVA IL REGISTRO C PER " + "
030 055 053             " +"
030 056 170             MOVAB     ACQUISISCE DI NUOVO I BIT DI STATO
030 057 346             ANI       CONTROLLA IL SEGNO
030 060 020             020
030 061 302             JNZ       SALTA SE IL SEGNO E' +
030 062 066             PLUS
030 063 030             0
030 064 014             INRC      SE LO E', AGGIUNGE 2
030 065 014             INRC
030 066 171    PLUS,    MOVAC     ACQUISISCE IL + O IL -
030 067 315             CALL      LO STAMPA
030 070 245             TTYOUT
030 071 030             0
030 072 315             CALL      CONTROLLA LA VIRGOLA DECIMALE
030 073 176             DECM
030 074 030             0
030 075 172             MOVAD     ATTIVA IL PRIMO CARATTERE
030 076 017             RRC
030 077 017             RRC
030 100 017             RRC
030 101 017             RRC
030 102 315             CALL      STAMPA IL NUMERO
030 103 210             NUMOUT
030 104 030             0
030 105 315             CALL      VERIFICA SE ESISTE UN ALTRO PUNTO
                                  DECIMALE
030 106 176             DECM
030 107 030             0
030 110 172             MOVAD     PRENDE IL NUMERO SUCCESSIVO
030 111 315             CALL      STAMPA IL NUMERO
030 112 210             NUMOUT
030 113 030             0
030 114 315             CALL      VERIFICA SE ESISTE L'ULTIMO PUNTO
                                  DECIMALE
030 115 176             DECM
030 116 030             0
030 117 173             MOVAE
030 120 017             RRC
030 121 017             RRC
030 122 017             RRC
030 123 017             RRC
030 124 315             CALL      STAMPA IL NUMERO SUCCESSIVO
030 125 210             NUMOUT
030 126 030             0
030 127 173             MOVAE     PRENDE L'ULTIMO NUMERO
030 130 315             CALL
030 131 210             NUMOUT
030 132 030             0
030 133 055    COUNT,   DCRL      SONO STATE STAMPATE CINQUE COLONNE?
030 134 312             JZ        SE SI', SALTA
030 135 155             NXTLIN
030 136 030             0
```

```
030 137 046            MVIH      SE NO, STAMPA 3 SPAZI
030 140 003            003
030 141 076   MRSP,    MVIA
030 142 040            040
030 143 315            CALL
030 144 245            TTYOUT
030 145 030            0
030 146 045            DCRH

030 147 302            JNZ
030 150 141            MRSP
030 151 030            0
030 152 303            JMP       DOPO 3 SPAZI ESEGUE UN'ALTRA
                                 ACQUISIZIONE
030 153 017            TIMER
030 154 030            0
```

QUESTA ROUTINE FA ESEGUIRE UN RITORNO DI CARRELLO E UN AVANZAMENTO DI UNA RIGA E RITORNA AL PUNTO DEL PROGRAMMA CHE INIZIALIZZA LE OPERAZIONI.

```
030 155 315   NXTLIN,  CALL
030 156 163            CRLF,
030 157 030            0
030 160 303            JMP
030 161 015            INIT
030 162 030            0
```

ESEGUE UN RITORNO DI CARRELLO E L'AVANZAMENTO DI 1 RIGA.

```
030 163 076   CRLF,    MVIA
030 164 015            015
030 165 315            CALL
030 166 245            TTYOUT
030 167 030            0
030 170 076            MVIA
030 171 012            012
030 172 315            CALL
030 173 245            TTYOUT
030 174 030            0
030 175 311            RET
```

SOTTOPROGRAMMA CHE ESEGUE IL TEST DEL BIT DI STATO DECIMALE
E METTE UN "." SE E' TROVATO A UNO.

```
030 176 170   DECM,    MOVAB     PRENDE LA PAROLA DI STATO
030 177 027            RAL       LA RUOTA DI UNA POSIZIONE
030 200 107            MOVBA     LA RIMETTE NEL REGISTRO B
030 201 330            RC        SE C'E' IL CARRY, RITORNA
030 202 076            MVIA      SE NON C'E', STAMPA "."
030 203 056            056
030 204 315            CALL
030 205 245            TTYOUT
030 206 030            0
030 207 311            RET
```

```
                              COSTRUISCE UN NUMERO ASCII PARTENDO
                              DAL VALORE BCD PRESENTE NEI 4 BIT MENO SIGNIFICATIVI
                              DEL REGISTRO A.

030 210 346    NMOUT,   ANI          MASCHERA I BIT NON VOLUTI
030 211 017             017
030 212 306             ADI
030 213 260             260
030 214 315             CALL

030 215 245             TTYOUT
030 216 030             0
030 217 311             RET

                        ROUTINE DI EMETTE "OVER" SE E' RICONOSCIUTA
                        UNA CONDIZIONE DI SUPERAMENTO DI CAPACITA'

030 220 345    OVER,    PUSHH
030 221 041             LXIH
030 222 261             MSG1
030 223 030             0
030 224 176    MLOOP,   MOVAM
030 225 376             CPI
030 226 000             000
030 227 302             JNZ
030 230 236             OUTPUT,
030 231 030             0
030 232 341             POPH
030 233 303             JMP          CONSIDERA QUESTA COME USCITA REGOLARE
030 234 133             COUNT        COSI' CHE INIZIA IL CONTEGGIO
030 235 030             0
030 236 315    OUTPUT,  CALL
030 237 245             TTYOUT
030 240 030             0
030 241 043             INXH
030 242 303             JMP
030 243 224             MLOOP
030 244 030             0

                        ROUTINE DI EMISSIONE SU TELESCRIVENTE.

030 245 365    TTYOUT,  PUSHPSW
030 246 333             IN
030 247 021             021
030 250 346             ANI
030 251 004             004
030 252 312             J2
030 253 246             TTYOUT + 1
030 254 030             0
030 255 361             POPPSW
030 256 323             OUT
030 257 020             020
030 260 311             RET

030 261 040    MSG1,    040
030 262 117             117
030 263 126             126
030 264 105             105
030 265 122             122
030 266 040             040
030 267 000             000
```

I quattro bit presenti inizialmente nelle posizioni D3-D0 nella parola di otto bit non sono ruotati, ma invece vengono mascherati con un'operazione di AND con il valore 00001111_2; e sommati al valore 10110000_2 per ricostruire l'equivalente ASCII. Un esempio può esservi d'aiuto per comprendere tutto ciò:

MMMMLLLL M = Cifra più significativa, 4 bit
L = Cifra meno significativa, 4 bit

Una volta che il dato è stato ruotato di 4 posizioni a destra, esso apparirà così:

LLLLMMMM

In seguito viene realizzata l'operazione di AND con il valore 00001111_2:

```
L L L L MMMM
0 0 0 0 1 1 1 1₂
------------------
0 0 0 0 MMMM = Risultato dell'operazione di AND
```

Ora, il valore 10110000_2 è aggiunto al risultato dell'operazione di AND per ottenere l'equivalente ASCII del carattere BCD, MMMM:

```
 0 0 0 0 MMMM
+1 0 1 1 0 0 0 0
------------------
 1 0 1 1 MMMM  = Equivalente ASCII della cifra BCD, MMMM
```

RILEVAZIONE DEL PUNTO DECIMALE

La posizione del punto decimale è determinata da un codice di uscita del pannello, codice rappresentato dalla presenza di un segnale su una delle tre uscite relative a questa funzione. Dal momento che il punto decimale può essere posto in ciascuna delle tre posizioni, ad esempio 0,1234 o 1,234 o 12,24 e la telescrivente stampa da sinistra a destra, è necessario eseguire il test per rilevare la presenza nella posizione più significativa. La posizione del punto decimale è indicata dalla presenza di un livello logico zero ad una delle tre uscite specifiche. L'uscita attivata fornisce l'informazione in accordo con le conversioni precedentemente descritte nel capitolo relativo alle uscite digitali dell'Esempio n.2 sull'interfacciamento di pannelli di misura digitali. L'informazione relativa al punto decimale è acquisita dal calcolatore tramite la porta di ingresso 202 ed ha la seguente rappresentazione:

```
0 0 0 X X X X X      Parola di stato e di flag
↑ ↑ ↑
↑ ↑ └Campo fino a 19,99 Volt
↑ └Campo fino a 1,999 volt
└ Campo fino a 0,1999 olt (199,9 millivolt)
```

Il programma memorizza questa parola di stato e relativa ai flag nel registro B dell'8080. Il sottoprogramma DECM è utilizzato in tre parti per eseguire il test sulla presenza del punto decimale e per effettuare la stampa del punto decimale se viene riconosciuto uno zero logico nella posizione corrente. Il sottoprogramma DECM ruota il flag-decimale nella posizione di carry dell'8080 e successivamente esegue il test tramite l'istruzione di ritorno in caso di carry, RC.

L'istruzioné RC provoca un ritorno al programma principale se non viene individuato nessun punto decimale nella apposita posizione durante la prova. Se, comunque il carry è posto a zero come risultato della rotazione, in modo da indicare che il "punto decimale" è

stato individuato, i passi di programma addizionali mettono in uscita un punto decimale su telescrivente in modo da riempire la porzione di carattere successivo della tensione, in fase di stampa. Dopo che il punto decimale è stato messo in uscita, il calcolatore restituisce il controllo al programma principale.

PROGRAMMA DI MESSAGGIO

Una delle esigenze del programma di acquisizione dati è quella di far scrivere al calcolatore "OVER" sulla stampante in caso di superamento della capacità. Il programma all'indirizzo simbolico OVER esegue la stampa puntando all'area di memorizzazione del messaggio, MSG1. I dati memorizzati qui e nelle cinque locazioni di memoria seguenti sono i caratteri ASCII:

MSG1	040	spazio
	117	O
	126	V
	105	E
	122	R
	040	spazio
	000	nullo

Una tipica uscita "in campo" occuperà sei posizioni per il segno, le quattro cifre e per un punto decimale. Anche il messaggio "OVER" occupa sei posizioni di carattere dato che abbiamo incluso le quattro lettere e i due spazi, uno prima del messaggio e uno dopo. L'ultimo carattere è un carattere nullo o tanti zeri. Questo segnala al programma OVER che è stata raggiunta la fine del messaggio in codice ASCII che deve essere emesso.

Il sottoprogramma OVER è di impiego generale in modo che possa essere utile alla vostra programmazione. È infatti molto più flessibile di un programma di uscita su telescrivente così realizzato:

```
MVIA       Carica il registro A con il dato
040        corrispondente al codice ASCII = spazio
CALL       Chiama il sottoprogramma di
TTYOUT     stampa su telescrivente
0
MVIA       Stampa un altro carattere
117
CALL
TTYOUT
0
.
.
ecc.
```

È anche molto più semplice cambiare il messaggio da emettere cambiando la stringa in codice ASCII dei caratteri memorizzati nella locazione di memoria successiva nell'area di memoria opportuna senza modificare la parte relativa al programma principale, che rimane di conseguenza immutato.

Queste ed altre tecniche di programmazione saranno il soggetto del futuro Bugbook, curato dal Dr. Christopher A. Titus. Esempi dettagliati e ben documentati sono riportati in questo prossimo volume. Il dott. Titus lavora presso la Tychon, Inc., Blacksburg, VA.

MODIFICHE DEI PROGRAMMI

Il programma di acquisizione dati presentato nell'Esempio 3-2 è adatto per molte appplicazioni, ma qualche utente può ritenere necessario di apportare delle modifiche in modo che si possa utilizzare in particolari situazioni. Riportiamo di seguito alcune delle modifiche più probabili che possono essere effettuate sul programma di acquisizione dati:

1) *I dati possono essere impaginati in un numero di colonne superiore.* Il numero di colonne dei dati è prefissato alzo simbolico INIT, tramite il dato che segue l'istruzione MVIL. Nel caso di 10 colonne il nuovo valore che dovrà seguire l'istruzione MVIL sarà 012. Si ricordi che il valore devere essere in codice ottale.

2) *Più spazi sono richiesti tra i valori dei dati.* Il numero degli spazi tra le colonne è stato prefissato nella parte del programma denomintata COUNT al valore tre tramite l'istruzione MVIH. Il numero degli spazi tra le colonne può essere modificato cambiando il byte immediatamente seguente questa istruzione MVIH.

Per eliminare del tutto gli spazi tra le colonne, è possibile sostituire un salto all'inidirizzo TIMER nei primi tre byte all'etichetta MRSP:

Passi di programma precedenti		Nuovi passi di programma	
MRSP,	MVIA	MRSP,	JMP
	040		TIMER
	CALL		0

Solo i primi tre byte alla locazione MRSP sono riportati in quanto solo questi sono quelli necessari di modifica per eliminare gli spazi tra le colonne relative ai dati in uscita. Si ricordi che dal momento che è stato inserito un salto incondizionato nel programma MRSP, i successivi byte, quelli vecchi "sia istruzioni che dati" non sono necessari di modifica in quanto non vengono interessati dal calcolatore. Questo è spesso chiamato una *pezza* nel programma. Alcuni programmi di elaborazione dati possono richiedere le eliminazioni degli spazi tra le colonne, in modo particolare se è necessario generare un nastro di carta per degli impieghi successivi.

3) *Cambiamento del messaggio in uscita.* Se non è di vostro gradimento il messaggio OVER che è emesso dal programma per indicare una condizione di superamento di capacità, è possibile cambiare il messaggio in codice ASCII specificando dei nuovi caratteri, a partire dalla locazione di memoria MSG1. Si ricordi che la fine del nuovo messaggio deve essere terminata con un carattere nullo, 000. È necesssario comunque, se si vuole mantenere l'allineamento, non superare i sei caratteri per il messaggio.

Alcuni programmi, come il BASIC o il FOCAL*, possono essere impiegati per elaborare i dati perforati sul nastro di carta del programma di acquisizione dati. Il formato deve essere evidentemente compatibile a questi programmi in modo che sia possibile l'interpretazione. Ciò significa che il messaggio "OVER" può provocare dei problemi in quanto il programma di elaborazione può interpretare i caratteri ASCII di ciascuna lettera come dei valori relativi alla grandezza acquisita. In questo caso il messaggio può essere cambiato in modo da consistere in una sola lettera, per esempio "S", per segnalare una condizione di superamento di capacità. Il valore più 0,000 può anch'esso servire per segnalare una condizione di superamento.

Alcuni programmi di elaborazione dati possono richiedere l'eliminazione del sottoprogramma che esegue il ritorno di carrello e di avanzamento di una linea poiché questi caratteri possono essere riconosciuti dal programma come caratteri terminatori del "file" di dati. La rimozione di questo sottoprogramma è facilmente effettuabile inserendo una istruzione di ritorno, RET, all'indirizzo simbolico CRLF.

Altri programmi ancora, possono richiedere un ritorno di carrello e un avanzamento di riga dopo ciascun valore dei dati. Questi ed altri cambiamenti sono facilmente realizzabili nel listato relativo al programma che abbiamo riportato e che potete modificare a vostro piacimento.

In questo capitolo abbiamo fornito le nozioni relative ai convertitori analogici-digitali del tipo a doppia pendenza. In seguito come passo logico successivo abbiamo introdotto i concetti relativi all'impiego di un pannello di misura digitale come convertitore A/D con uscite in codice BCD. Avrete notato che questo dispositivo è particolarmente utile in quelle applicazioni che richiedono un'indicazione visiva del segnale (o grandezza) che deve essere misurata dal calcolatore. Uscite di questo tipo possono essere particolarmente comprensibili da parte di operatori umani. Abbiamo anche fornito due esempi di come i dati in codice BCD possono essere memorizzati e utilizzati.

**FOCAL è un marchio registrato dalla Digital Equipment Corporation, Maynard, Massachusetts.*

CAPITOLO 4

MISCELLANEA DI TECNICHE DI CONVERSIONE

INTRODUZIONE

Ci sono un numero di tecniche di conversione usate con i convertitori analogico-digitali accanto a quelle precedentemente descritte come quella a rampa, a successive approssimazioni e a doppia pendenza. Queste tecniche addizionali non sono di impiego comune, ma possono giustificare il loro impiego in particolari situazioni e riteniamo che sia importante che ne acquisiate un minimo di familiarità prima di affrontare i nuovi argomenti.

CONVERTITORI TENSIONE-FREQUENZA

I convertitori tensione-frequenza non sono così largamente impiegati come le altre tecniche discusse in precedenza, cioé le conversioni a rampa, ad approssimazioni successive e a doppia pendenza. Essi trovano applicazione in sistemi con dinamiche di variazione elevate, generalmente di alcune decadi, dove le conversioni possono essere lente e dove i sensori che generano i valori di tensione in uscite possono essere molto distanti dal dispositivo di misura o dal calcolatore.

I convertitori tensione-frequenza operano convertendo una tensione in ingresso in una frequenza in uscita, dove quest'ultima è proporzionale al valore della grandezza in ingresso. La frequenza in uscita è generalmente compatibile con valori TTL o CMOS. L'uscita può essere inoltre rappresentata sia da una serie di impulsi o da un'onda quadra ad una frequenza generata dal particolare valore in ingresso. La linearità per i convertitori V/F è generalmente dello 0,05% in specifiche condizioni di funzionamento e relativamente al campo di variazione della temperatura del particolare dispositivo impiegato.

Un tipico convertitore tensione-frequenza sono i circuiti della serie VFV della Datel Systems, Inc., e l'AD537 della Analog Devices. Ciascuno di questi dispositivi ha una frequenza massima di tenuta che è tra i 100 e i 150 kHz, e possono inoltre operare come convertitori frequenza-tensione per ricostruire il valore di tensione da misurare. Il circuito integrato della Analog Devices può anche operare in modo bipolare, così che è possibile effettuare misure di tensione sia positive che negative. Un alimentatore a doppia uscita o due alimentatori separati sono necessari per questo modo di funzionamento. Nella Figura 4-1 sono riportati sia il VFV che l'AD537.

La misura dei segnali analogici viene effettuata semplicemente contando il numero di impulsi o di transizioni di un oscillatore in un'unità di tempo, generalmente un secondo. Un semplice contatore decimale a 4 o 5 cifre può essere impiegato per visualizzare il risultato di una conversione; oppure si può utilizzare un contatore binario se questo tipo di conteggio è più indicato, come può essere nel caso di interfacciamento con calcolatori. I tempi di conversione sono generalmente lunghi in quanto il conteggio deve essere completato. Frequenze di conversione che permettono 10 misure al secondo sono possibili, con tempi di misura più lunghi che possono andare da 1 a 10 secondi a seconda che venga richiesta una risoluzione maggiore.

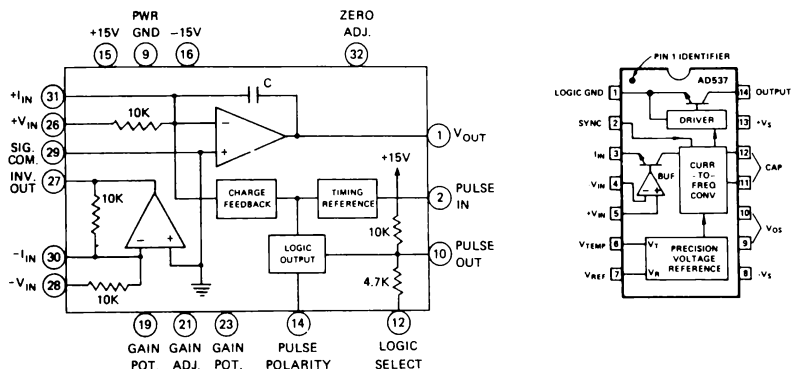



Figura 4-1. Tipici dispositivi di conversione tensione-frequenza. Il circuito della serie VFV della DATEL è riportato sulla sinistra, mentre a destra è rappresentato il circuito integrato AD537 della Analog Devices. Per gentile concessione della Datel Systems, Inc.

Anche se i convertitori V/F sono lenti se comparati alla velocità di conversione dei convertitori ad approssimazioni successive, essi sono frequentemente impiegati dove è necessario integrare un segnale su un periodo di tempo relativamente lungo; infatti eseguono conversioni in modo che l'integrazione produca un computo totale relativo al periodo di tempo interessato.

L'uso dei convertitori V/F in circuiti di integrazione fornisce una semplice soluzione per misurare l'area relativa al picco in uscita da uno strumento. Dal momento che un'integrazione realizza una somma relativa ad un periodo di tempo, segnali di disturbo periodici sovrapposti alla tensione da misurare può essere integrata a zero.

L'interfacciamento dei convertitori V/F ai microcalcolatori è molto semplice dal momento che è necessario solo un contatore. La tecnica di conversione è particolarmente utile quando le tensioni da misurare o digitalizzare sono molto distanti dal calcolatore; in

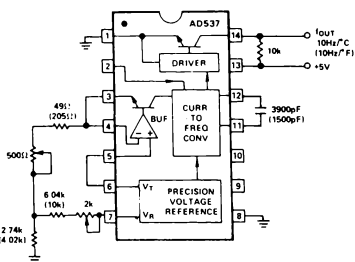



Figura 4-2. Uso del convertitore V/F della Analog Devices come sensore di temperatura. Il segnale in uscita cambia a una frequenza di 10 Hz quando i componenti passivi assumono i valori riportati nella figura. I componenti permettono una misura in gradi Celsius o Farenheit. Per gentile concessione della Analog Devices, Inc.

questo caso solo due fili sono necessari per trasmettere il valore di frequenza al calcolatore da un posto di misura remoto. L'impiego di circuiti "line driver e receiver" rende questa tecnica particolarmente immune ai disturbi.

Il convertitore tensione-frequenza AD537 dell'Analog Devices è un dispositivo particolarmente interessante per il suo basso costo e il largo campo di impiego. Esso può essere interfacciato con i più comuni circuiti, può essere "multiplexato" e può operare in una configurazione con due soli fili dove anche l'alimentazione è ottenuta tramite i collegamenti che permettono il trasferimento del segnale in frequenza. Questo è un particolare importante nel caso di applicazioni di acquisizioni remote, dal momento che ciascun sensore non richiede localmente un alimentatore personale.

L'AD537 ha un'unità lineare, proporzionale alla temperatura che permette l'impiego di questo circuito integrato come sensore di temperatura, convertendo questa grandezza in una frequenza. Un circuito tipico è riportato nella Figura 4-2.

Come precedentemente accennato quando un convertitore V/F è impiegato nell'interfacciamento con un calcolatore, si può utilizzare sia un contatore decimale che binario per calcolare il valore di frequenza. In alcune applicazioni è possibile l'uso di entrambi. Nel circuito riportato in Figura 4-2, un contatore decimale può essere usato per una diretta lettura dei valori mentre un contatore binario è usato *nello stesso tempo* per fornire al calcolatore il valore binario della temperatura.

CONVERTITORI FLASH

La tecnica conversione *"flash" o simultanea*, funziona comparando un valore di tensione sconosciuto che deve essere digitalizzato con una serie di potenziali di riferimento

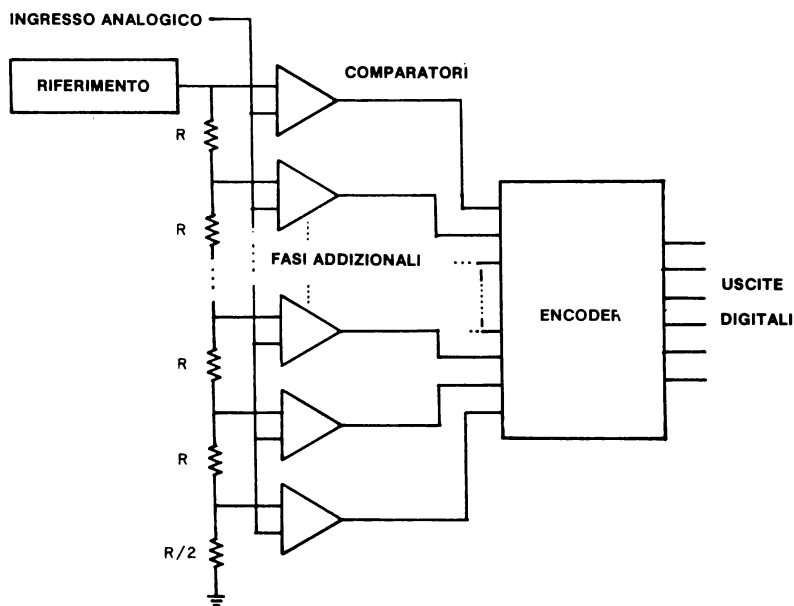



Figura 4-3. Schema a blocchi di un tipico convertitore "flash". La sezione "encoder" converte le uscite dei comparatori in un codice binario a sei bit.

prefissati. La comparazione può aver luogo simultaneamente; così per una conversione a n-bit sono necessarie 2^n-1 confronti. Per una conversione di 8 bit, sarebbero necessari 255 comparatori e punti di riferimento. Viene impiegato un solo stabile punto di riferimento con una serie di resistenze che forniscono i valori di tensione necessari per i confronti. L'ingresso di riferimento del primo comparatore è posizionato ad un valore che è la metà del minimo incremento, mentre tutti gli altri ingressi vengono posti ad un valore di tensione superiore a quello precedente pari ad un minimo di incremento. Lo schema a blocchi di un convertitore simultaneo è riportato in Figura 4-3.

Le uscite fornite dai comparatori devono essere decodificate in qualche modo per ottenere un'uscita binaria che rappresenti il valore di tensione sconosciuto che deve essere misurato. Il circuito può diventare complesso per convertitori che hanno sette o più bit.

Dal momento che viene usato un comparatore per ciascun punto di riferimento, questa tecnica di conversione è utile solo per conversioni veloci di 4, 5 o 6 bit. Il tempo di conversione è limitato solo dal tempo di risposta dei comparatori e dalla velocità dei circuiti logici che realizzano la conversione del codice.

Una combinazione di tecniche di conversione flash e convertitori D/A con reazione viene utilizzata per incrementare la risoluzione possibile a più di otto bit, ma le velocità di conversione sono più lente e i costi più alti di una soluzione a sei bit. Quando un convertitore D/A è impiegato con compiti di reazione in convertitori flash, esso viene chiamato "*convertitore con reazione simultanea*" o "*simultaneas riffler converter*". Lo schema a blocchi di questo tipo di convertitore è riportato in Figura 4-4.

Il convertitore con reazione simultanea funziona utilizzando due convertitori "flash" a quattro bit simili a quello riportato nella Figura 4-3 che nella Figura 4-4 sono indicati come uno e due (1 e 2). Il valore di tensione in ingresso da misurare è prima digitalizzato in un codice con risoluzione di quattro bit usando il convertitore 1. Per un campo di variazione da zero a un volt, ciò significa una risoluzione di una parte su 16 o 15 porzioni di 66,7 millivolt ciascuna.

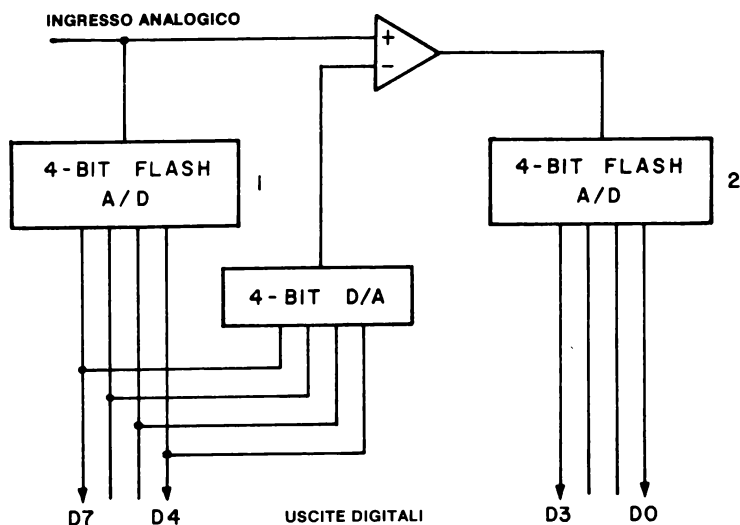



Figura 4-4. Schema a blocchi di un convertitore con reazione simultanea. Due convertitori "flash" a quattro bit sono necessari per questo tipo di schema.

Noi assumeremo che il valore da misurare sia di 690 millivolt ed esamineremo il modo di funzionamento del convertitore con reazione simultanea riportato in Figura 4-4. Il convertitore può convertire il segnale sconosciuto solo in sedici possibili codici in uscita rappresentati i quindici valori compresi tra zero e un volt, ognuno superiore al precedente dei 66,7 millivolt di cui si è parlato prima. I valori di tensione possibili relativi ai singoli codici prodotti dal convertitore sono riportati nella tabella nel seguito. *Ricordatevi che il primo comparatore ha un potenziale di riferimento pari a metà del minimo incremento e che tutti gli altri comparatori hanno il loro potenziale di riferimento superiore di un valore pari al minimo incremento rispetto a quello precedente.*

Codici	Tensione (mv)
15	967
14	900
13	833
12	767
11	700
10	633
9	566
8	500

Codici	Tensione (mv)
7	433
6	366
5	300
4	233
3	166
2	100
1	33

L'ingresso da misurare che abbiamo supposto di 690 millivolt cade tra i codici 10 e 11. I comparatori da uno a dieci indicano che la tensione da misurare è superiore ai propri valori di riferimento come si può dedurre dalla tabella, così il codice corrispondente all'ultimo comparatore che rileva una condizione di superamento del valore di riferimento diventa il codice che rappresenta il valore sconosciuto. In questo caso è $10 = 1010_2$. *Notate che non esiste il codice relativo al valore zero. Se nessuno dei comparatori indica un superamento la logica di decodifica emetterà un codice* 0000_2.

Il valore di tensione usato come riferimento corrisponde al decimo comparatore è solo 633 millivolt, così che rimane una differenza di 57 millivolt tra i due valori, di riferimento e di misura, eccessiva per rendere significativa la misura.

Il codice binario prodotto dal primo convertitore "flash" a quattro bit è collegato agli ingressi di un convertitore digitale analogico che è predisposto per generare una tensione uguale al potenziale di riferimento dell'ultimo comparatore che ha indicato una condizione di riferimento. In questo caso è il comparatore 10 che fa in modo di far mettere in uscita al convertitore "flash" un valore di 633 millivolt.
Questo valore è sottratto dal valore di tensione sconosciuto utilizzando un amplificatore differenziale, la cui uscita è uguale quindi alla differenza tra i due valori, quello da misurare e quello del potenziale di riferimento dell'ultimo comparatore che indica superamento.

In questo caso la differenza è:

690 millivolt del segnale da misurare
−633 millivolt del riferimento del comparatore
57 millivolt di differenza.

I 57 millivolt di differenza sono uguali alla porzione di valore che rendono la misura non accettabile come abbiamo detto prima. Il secondo convertitore "flash" a quattro bit è predisposto in modo da avere una esecuzione massima di 66 millivolt, che è il massimo valore che può rimanere irrisolto dalla prima conversione.

In questo tipo di conversione, un primo dispositivo esegue una conversione del segnale da misurare al meglio di 4 bit di risoluzione. La differenza tra questo valore ottenuto viene

presentata ad un secondo dispositivo dopo una rete composta da un convertitore D/A e un dispositivo che esegue una differenza.

Dal momento che dei convertitori flash vengono impiegati in questi casi, è possibile che siano necessari molti comparatori e il loro numero 'Q' può essere calcolato nel seguente modo:

$$Q = 2^{\{(n/2)+1\}} - 2 \quad n = \text{numero di bit di risoluzione}$$

Per la conversione a otto bit illustrata nella Figura 4-4, sono necessari 30 comparatori. È importante notare che anche se è stato impiegato un convertitore a 4 bit, deve avere la precisione almeno di otto bit o di una parte su 256 per mantenere la precisione del sistema nel suo complesso.
Così come tutti i componenti in questo tipo di sistema devono avere una precisione dello stesso ordine della risoluzione richiesta all'intero sistema. In caso contrario si possono verificare errori significativi nel processo di digitalizzazione.

Un ulteriore affinamento di questo convertitore con reazione simultanea è *la tecnica di conversione flash con soglia variabile*, ma è al di fuori del nostro scopo la sua discussione; per cui vi rimandiamo, nel caso di vostro interesse, al *Motorola Application Note AN-702* dove l'argomento è affrontato in modo approfondito.

L'unico altro tipo di convertitore, di cui parleremo, è il *convertitore "a traccia".* Esso può essere immaginato come un convertitore A/D veloce a rampa che può generare tramite l'uso di contatori bidirezionali di una rampa positiva e una negativa. Il convertitore insegue continuamente la tensione di ingresso tramite l'uscita di un comparatore, come per un normale convertitore a rampa, per indicare quale deve essere l'andamento della rampa (in sù o in giù) per raggiungere il valore sconosciuto. Nello stesso tempo le uscite digitali sono sempre attivate, per indicare il corrente valore digitale dell'ingresso da misurare.

Ci sono probabilmente altre tecniche di conversione, ma non sono direttamente applicabili ai calcolatori. La discussione affrontata relativa alle tecniche a rampa a doppia pendenza, approssimazioni successive e tensione-frequenza probabilmente copre il 95% delle tecniche oggi in uso. Convertitori ad approssimazioni successive più veloci sono stati annunciati e potranno raggiungere la velocità di alcuni convertitori "flash".

CAPITOLO 5

CIRCUITI DI SAMPLE-AND-HOLD E DISPOSITIVI MULTIPLEXER

INTRODUZIONE

Lo scopo di questo capitolo è di introdurvi all'uso di circuiti sample-and-hold e di multiplexer nei sistemi analogici. Noterete che i circuiti di sample-and-hold sono particolarmente utili quando si vogliono misurare segnali variabili utilizzando convertitori A/D.
I multiplexer sono molto spesso impiegati in applicazioni in cui viene usato un solo convertitore A/D per acquisire dei segnali da più sensori.

CIRCUITI SAMPLE-AND-HOLD

I dispositivi di sample-and-hold (S/H) o gli amplificatori che racchiudono questo tipo di circuito (SHA) sono elementi di circuiti analogici equivalenti ai circuiti latch nel campo digitale, e vengono usati quando si desidera campionare un segnale analogico e mantenerlo fisso ad un particolare valore in modo che la grandezza relativa possa essere misurata o comunque utilizzata dal sistema. Il funzionamento ideale di un dispositivo S/H è illustrato nella Figura 5-1. In questo esempio, l'uscita dell'S/H segue o riproduce la tensione di ingresso per tutto l'intervallo di *campionamento* e mantiene costante l'ultimo valore di tensione analogico dal momento in cui si commuta nel modo *hold*. Per maggiore chiarezza, nella Figura 5-1, le curve relative alle tensioni di ingresso e di uscita sono state spostate leggermente tra di loro.

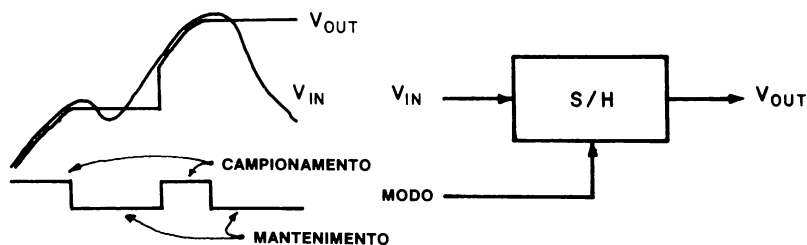



Figura 5-1. Segnali di ingresso e di uscita per un dispositivo S/H ideale per entrambi i modi di funzionamento.

Il più semplice circuito S/H che si possa realizzare è quello costituito da un condensatore e da un interruttore. Per i condensatori si consiglia di utilizzare quelli a bassa dispersione come quelli al polistirene o al ci teflon. Questo primo esempio di circuito è illustrato nella Figura 5-2.

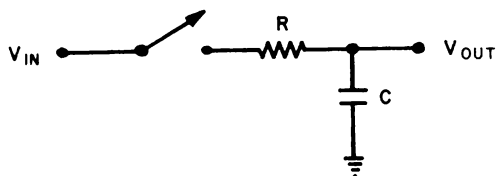



Figura 5-2. Impiego di un condensatore e di un interruttore per la realizzazione di un circuito sample-and-hold.

In questa prima realizzazione di circuito S/H il condensatore continuerà a caricarsi o a scaricarsi mantenendo V_{OUT} uguale a V_{IN} una volta che l'interruttore sia chiuso. La velocità di carica del condensatore sarà stabilita dal valore della resistenza e ciò farà in modo che la tensione presente in uscita, V_{OUT} non seguirà immediatamente quella di ingresso. Nel nostro caso, abbiamo supposto che un generatore di tensione sia in grado di fornire la corrente di carica richiesta dalla rete resistore-condensatore. Il resistore nello schema può avere valori molto bassi e può considerarsi al limite come la resistenza introdotta dal commutatore e dai collegamenti. Bisogna inoltre ricordare che il carico sull'uscita del condensatore tenderà a scaricare il condensatore stesso, diminuendo il valore di tensione.

La maggior parte dei circuiti S/H sono più complessi in quanto utilizzano amplificatori operazionali con l'ingresso ad alta impedenza e transistori ad effetto di campo (FET). Un circuito tipico è riportato nella Figura 5-3.

Il commutatore che controlla il modo, riportato nella Figura 5-3 è un commutatore a *ponte di diodi* e a transistor FET. Per maggiori dettagli sui dispositivi S/H sia per quanto riguarda la loro progettazione che le caratteristiche di funzionamento vi rimandiamo al testo della Siliconix Incorporated, *Analog S vitches and their Applications*[4].

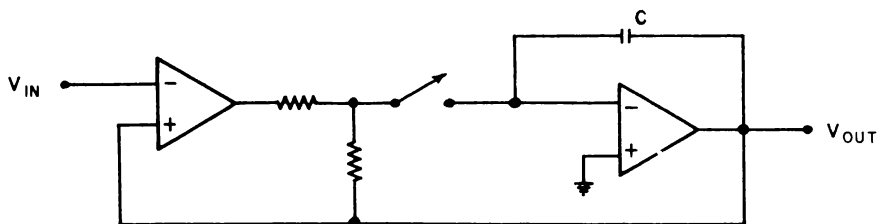



Figura 5-3. Tipico circuito sample-and-hold basato su un amplificatore operazionale. Possono essere usati amplificatori con ingresso ad alta impedenza (FET).

USO DEI DISPOSITIVI SAMPLE-AND-HOLD

I dispositivi S/H hanno diverse funzioni nelle applicazioni che impiegano convertitori D/A e A/D, e possono essere utilizzati per:

- Mantenere costante un segnale analogico in modo da permettere una conversione A/D.
- Campionare contemporaneamente molti ingressi analogici per una successiva misura (in questo caso è necessario un dispositivo S/H per ogni ingresso).
- Togliere le impurità di un segnale in uscita di un convertitore D/A.
- Distribuire la tensione in uscita di un convertitore D/A a più punti in cui è necessario mantenere il valore costante.

Il secondo e il quarto degli impieghi presentati sono diventati meno importanti di quanto non lo fossero due o tre anni fa, dal momento che probabilmente oggi è meno costoso riservare un convertitore A/D per ogni ingresso e un convertitore D/A per ogni uscita.

I dispositivi S/H sono frequentemente usati per *"pulire"* i segnali in uscita dei convertitori D/A, fornendo una *doppia bufferizzazione analogica*. Una volta che il convertitore ha eseguito la conversione e la sua uscita si è stabilizzata in un ambito di errore accettabile, viene campionata e mantenuta costante dal dispositivo S/H, la cui uscita è presentata al dispositivo normalmente connesso al convertitore D/A. Questa configurazione è particolarmente utile nei casi in cui sia richiesta una dinamica veloce al convertitore, ma nello stesso tempo non possano essere tollerati disturbi dell'assestamento (glitches). Segnali per CRT che impiegano dei convertitori D/A per comandare il fascio elettronico vengono spesso trattati in questo modo.

L'uso più comune dei dispositivi S/H è quello di campionare e mantenere costante un segnale analogico mentre viene misurato da un convertitore e si dimostra particolarmente utile nel campo della strumentazione quando l'uscita di uno strumento varia troppo rapidamente per essere misurata da un convertitore A/D con la tecnica delle approssimazioni successive o con quella a rampa. Si ricordi, a questo proposito, l'esempio riportato nel Capitolo 2, in cui si ripropone di rendere digitale un'uscita di picco con dei convertitori controllati da software con i metodi citati.

Quando un dispositivo S/H viene utilizzato all'ingresso di un comparatore di un segnale da misurare (per esempio nel caso di un comparatore A/D controllato da programma) o in un circuito di conversione A/D, una digitalizzazione può dare una rappresentazione precisa della tensione di picco da misurare. Lo schema a blocchi della Figura 5-4, indica che ciò può essere effettuato.
Il segnale proveniente dal calcolatore permette al circuito S/H di campionare il segnale sconosciuto. Quando viene raggiunto il valore di picco, lo strumento genera il segnale opportuno, portando il dispositivo nel modo Hold.

Il diagramma dei tempi, illustrato nella Figura 5-5, riporta le relazioni temporali tra i segnali impiegati per controllare il dispositivo S/H della figura precedente. Viene inoltre riportata l'uscita analogica dello strumento così come l'uscita del dispositivo S/H.

A questo punto si potrebbe impiegare o un convertitore A/D a rampa lenta o uno veloce, ad approssimazioni successive, per fornire l'esatto valore digitale del picco di tensione presente in uscita dallo strumento.
Il dispositivo S/H svolge la funzione di mantenere la tensione costante in modo da poterla misurare.

L'uso di un dispositivo S/H all'ingresso di un convertitore A/D aumenta sensibilmente la sua capacità di misurare segnali variabili. Se non si utilizzasse questo dispositivo anche i convertitori più veloci non potrebbero essere utilizzati se non per misurare segnali che

variano molto lentamente. Per esempio, la frequenza massima di un'onda sinusoidale che può essere misurata con un convertitore A/D a 10 bit, che operi a una velocità di conversione di 10 microsecondi e una precisione di ± 1 LSB, è di 16 Hz[2], se si presuppone che il convertitore A/D stia effettuando la misura di un segnale di ampiezza pari al valore di fondo scala.

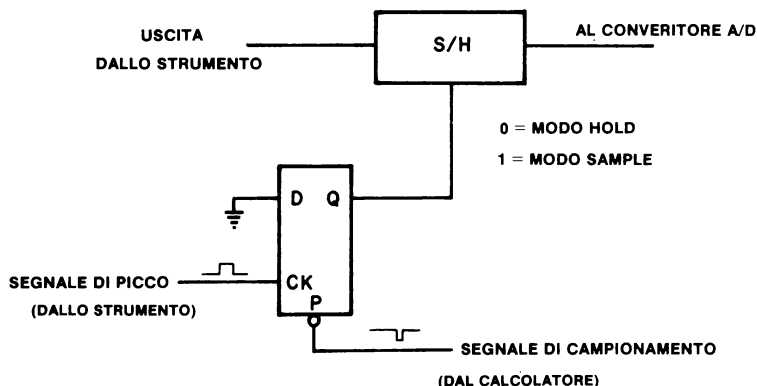



Figura 5-4. Schema a blocchi di un tipico interfacciamento tra un dispositivo sample-and-hold e un calcolatore.

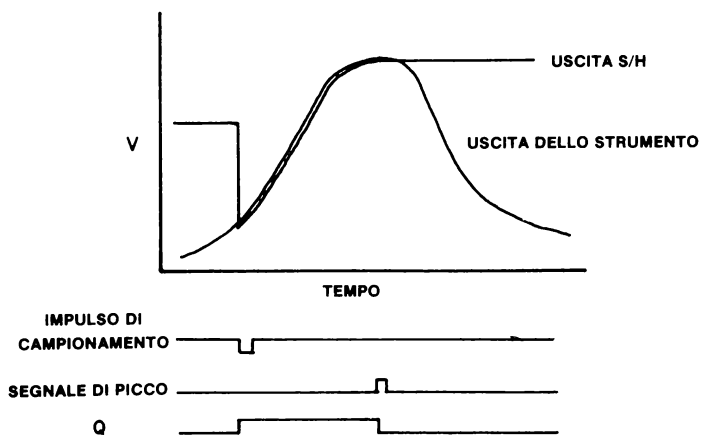



Figura 5-5. Tipico diagramma dei tempi di un circuito S/H controllato dallo strumento e dal calcolatore. I segnali di ingresso e di uscita sono riportati leggermente spostati per maggiore chiarezza. Per quanto riguarda lo schema di interfacciamento si rimanda alla Figura 5-4.

Possiamo esprimere matematicamente che la relazione frequenza/conversione:[2]

Per una precisione di $\pm$ 1/2 LSB la frequenza di variazione massima è:

$$\frac{\Delta V}{\Delta t} \leqslant \frac{1}{2} \; \frac{\text{tensione di fondo scala}}{2^n}$$

che corrisponde a:

$$\frac{\Delta V}{\Delta t} \leqslant \frac{\text{tensione di fondo scala}}{2^{n+1}}$$

Così per un convertitore che ha un valore di fondo scala di 10 volt la frequenza di variazione deve essere inferiore a 5 millivolt in 10 microsecondi o 500 Volt in un secondo.

Per calcolare la frequenza per un particolare campo di tensione può essere utilizzata la seguente formula:

$$\frac{\Delta V}{\Delta t} = (\text{campo di variazione di ingresso}) \cdot (2\, \pi\, f)$$

In questa formula il campo di variazione di ingresso è di solito il valore di fondo scala e la frequenza f è la frequenza massima della velocità di variazione.

Questi calcoli dovrebbero dimostrare l'utilità dei dispositivi S/H e la ragione per cui vengono spesso utilizzati con i convertitori A/D. Nell'esempio di acquisizione dati precedentemente trattata in cui erano state misurate le temperature di una serra, queste ultime probabilmente non subivano variazioni abbastanza rapide da richiedere l'uso di un dispositivo di questo genere. In molte altre situazioni l'impiego dei dispositivi S/H è invece una norma.

I circuiti di S/H non hanno evidentemente un funzionamento ideale. Ci sono alcuni termini che vi aiuteranno a capire meglio i loro limiti e i campi di impiego; li riportiamo qui di seguito, facendo riferimento alle tre Figure 5-6, 5-7, 5-8.

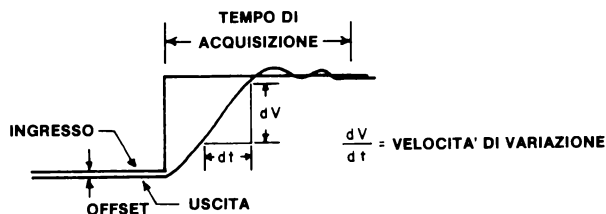



Figura 5-6. Rappresentazione del tempo di acquisizione, offset, e velocità di variazione.

<u>D.C. OFFSET</u> Rappresenta la differenza tra i valori in ingresso e in uscita del dispositivo quando l'ingresso è posto a terra e viene di solito espresso in millivolt. L'offset può essere regolato mediante componenti esterni, ma di regola subisce delle variazioni in funzione del tempo e della temperatura durante il funzionamento.

TEMPO DI ACQUISIZIONE Il dispositivo di S/H non ha un funzionamento ideale e richiede un certo tempo tra l'acquisizione effettiva e il momento in cui il circuito effettua il campionamento una volta che si trova nel modo *sample*; il tempo di acquisizione è il tempo che intercorre tra il momento in cui viene effettuato il campionamento e il momento in cui viene fornita l'informazione che rimane all'interno di una variazione dello 0,01% rispetto al valore di ingresso. Il tempo di acquisizione è normalmente di pochi microsecondi e può essere definito come *il tempo di mantenimento* quando il dispositivo si trova nel modo sample.

VELOCITA' DI VARIAZIONE Si definisce normalmente la velocità massima di variazione dell'uscita, che è espressa in Volt/secondo, ed è una limitazione imposta dalla velocità di carica dei condensatori che svolgono la funzione di mantenimento della tensione e dai tempi di risposta degli amplificatori operazionali utilizzati nel circuito.

TEMPO DI APERTURA È il periodo di tempo richiesto dal dispositivo per andare dal modo sample al modo hold ogni volta che viene ricevuto il comando di cambiamento di stato. Quest'ultimo non avviene mai istantaneamente.

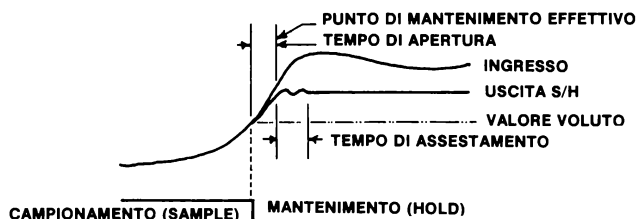



Figura 5-7. Rappresentazione tempo di apertura, e dell'incertezza in apertura. Nel disegno viene riportato anche il tempo di assestamento.

INCERTEZZA IN APERTURA Rappresenta la variazione del tempo di apertura tra un dispositivo e l'altro. È normalmente limitata a pochi ns.

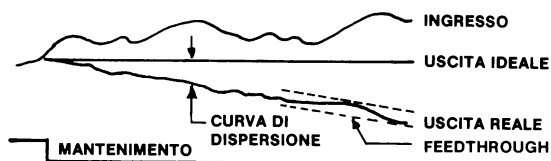



Figura 5-8 Rappresentazione della linearità, della velocità di deriva e del feedthrough.

LINEARITA' La linearità rappresenta la variazione dell'uscita rispetto al valore previsto riguardo all'intera gamma di tensioni che l'uscita può assumere. Questa differenza tra l'ingresso reale e ingresso ideale è riportata nella Figura 5-8. Può anche essere definita come un *errore di guadagno* ed essere quindi espressa in percentuale. Ad

esempio, 0,01%, oppure come la deviazione nel diagramma ingresso/uscita reale, da una linea rettilinea ottimale.

VELOCITA' DI DERIVA Dal momento che il circuito S/H è costituito da un condensatore che accumula carica elettrica, si avrà nel tempo una tendenza da parte del condensatore stesso a scaricarsi.
La variazione del livello di carica del condensatore è in funzione della corrente di dispersione del commutatore e della corrente richiesta dagli elementi del circuito collegati.
Questa grandezza chiamata anche *variazione di tendenza* è espressa in millivolt per secondo.

FEEDTHROUGH Indica di quanto si discosta il valore in ingresso da quello messo in uscita quando il dispositivo S/H è nel modo hold e può venire espresso in percentuale o in *decibel* (dB).

LARGHEZZA DI BANDA PER PICCOLI SEGNALI Fornisce la frequenza alla quale l'uscita dell'S/H viene attenuata di 3 dB (circa il 30%) rispetto al valore di ingresso. Questa caratteristica non è riportata nelle figure. Le Figure 5-6, 5-7 e 5-8 sono dedotte da quelle riportate nel *Analog Digital Conversion Handbook*[1].

Ci sono dei limiti del funzionamento dei dispositivi S/H.
Quelli che hanno brevi tempi di acquisizione, infatti, usano piccoli condensatori e di conseguenza avranno un'alta velocità di deriva. L'uso di condensatori più grandi comporta dei tempi di acquisizione maggiori anche se si avrà un mantenimento più lungo del valore di tensione di carica.
Quando in un'applicazione sono richieste acquisizioni veloci e tempo di mantenimento lunghi, possono essere usati due circuiti S/H. Il primo servirà ad acquisire in modo rapido il segnale analogico al punto che interessa, il secondo, collegato all'uscita del primo servirà a mantenere stabile il segnale che può acquisire in un tempo superiore. Un esempio di questi dispositivi S/H in cascata è illustrato nella Figura 5-9.

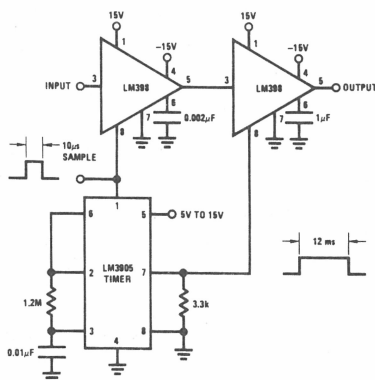



Figura 5-9. Uso dei dispositivi S/H in una configurazione a cascata che permette un'alta velocità di acquisizioni e una bassa dispersione. Per gentile concessione della National Semiconductor Corporation.

Occorre ricordare che in questo tipo di configurazione a cascata la velocità di campionamento complessiva sarà determinata dalla somma delle velocità di entrambi i dispositivi.

Esiste una vasta gamma di dispositivi S/H disponibili sul mercato che permette di semplificare i progetti di circuiti analogici.

Analog Devices, Inc., Norwood, MA 02062

SHA-5	General Purpose
SHA-1A	General Purpose

Burr-Brown Research Corp., Tucson, AZ 85734

SHC80KP	Low Cost
SHM60	High Speed

Datel Systems, Inc., Canton, MA 02021

SHM-LM2	Low Cost, Integrated Circuit
SHM-CM	General Purpose, ± 12 volt range

Hybrid Systems Corp., Bedford, MA 01730

SH703	Low Cost

Intersil, Inc., Cupertino, CA 95014

IH5110	Low Cost, Integrated Circuit

National Semiconductor Corp., Santa Clara, CA 95051

LF389	Low Cost, Integrated Circuit

Teledyne-Philbrick, Dedham, MA 02026

Model 4853	High Speed

MULTIPLEXERS

In molte applicazioni che impiegano convertitori A/D, diventa troppo costoso dedicare un convertitore per ogni sensore. Una soluzione alternativa è quella di utilizzare un solo convertitore A/D per più sensori; questa tecnica è definita *"multiplexing"* dal momento che molte sorgenti di segnali si dividono una linea comune di collegamento con un singolo dispositivo di ricezione, nel nostro caso il convertitore.

Un multiplexer può essere semplice come un commutatore rotante a più posizioni o complesso come un multiplexer per comunicazioni a microonde usato dalle compagnie telefoniche. In ogni caso la possibilità di comunicazione è riservata ad un segnale per un breve periodo di tempo, prima che avvenga la commutazione al successivo canale che deve essere collegato. In questo modo uno e uno solo può essere il canale attivato in ogni singolo momento. Un multiplexer può essere quindi pensato come un semplice commutatore, come illustrato nella Figura 5-10.

I multiplexer assumono notevole importanza quando si parla di convertitori A/D, ma il nostro scopo in questo paragrafo non è di fornire una loro descrizione dettagliata, ma di presentare alcuni casi di impiego con particolare riferimento a quelle applicazioni che

richiedono la necessità di smistare un certo numero di segnali analogici ad un solo convertitore A/D.
Per maggiori particolari su questi dispositivi e le loro caratteristiche di funzionamento, vi rimandiamo a due testi:
Data Conversion Handbook[2] e *Analog Switches and their applications*[4]. Tra le specifiche tecniche dei singoli multiplexer fornite dai costruttori, troverete anche informazioni relative ad applicazioni specifiche.

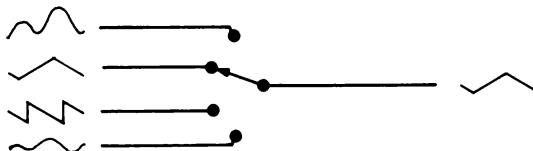



Figura 5-10. Semplice multiplexer realizzato con un commutatore rotante che mostra quattro possibili segnali in ingresso, ognuno dei quali può essere commutato in uscita.

MULTIPLEXER ELETTROMECCANICI

I multiplexer analogici più semplici sono costituiti da commutatori meccanici che possono essere azionati manualmente per ottenere il necessario incanalamento dei segnali. Questa soluzione di commutazione non è molto conveniente per quelle applicazioni di acquisizione dati nelle quali si debba commutare i vari canali con una certa velocità.

I relè rappresentano un valido compromesso dal momento che sono piccoli e facili da attivare con segnali a livello TTL o con dispositivi come il circuito integrato composto da due driver di periferica SN75451/A. Alcuni relè di tipo reed per piccoli segnali sono disponibili nel formato dual-in-line, lo stesso dei circuiti integrati a 14 e 16 piedini, e sono generalmente utilizzati con segnali di basso livello dal momento che sono compatti e hanno una bassa resistenza di contatto.

I relè meccanici producono un effetto di rimbalzi all'apertura e chiusura dei contatti che appare al circuito come una serie di brevi e continue aperture e chiusure dei contatti stessi. Anche se il periodo del rimbalzo dei contatti è breve, pochi millisecondi, può introdurre disturbi nel segnale analogico se viene misurato in questo istante. Il passaggio di piccoli valori di corrente non può produrre la "pulizia" della superficie metallica di contatto come succede nel caso di passaggio di forti correnti, così che la resistenza del contatto diviene sempre maggiore con il passare del tempo. Relé con contatti a bagno di mercurio permettono invece il collegamento di segnali a basso livello utilizzando appunto il mercurio per realizzare un contatto "pulito", a bassa resistenza, metallo su metallo, che non si deteriora con il passare del tempo.

I relé reed sono disponibili in una varietà di configurazioni come: un polo a singolo contatto (SPST), un polo a doppio contatto (SPDT) e così via. Generalmente il tipo SPST viene utilizzato per multiplexare un segnale analogico singolo. Il costo dei relé reed va dai 3 ai 20 Dollari a seconda della qualità e della configurazione dei contatti richiesta.

I dispositivi meccanici hanno delle limitazioni che devono essere tenute in considerazione come l'usura meccanica dei contatti, dal momento che si trovano sottoposti a pressione ad ogni chiusura del contatto e di conseguenza hanno un periodo di vita limitato. Si consideri che un relé reed meccanico con contatti a secco ha una vita minima pari a 100 milioni di operazioni. Ciò significa che, ad una velocità di 10 campionamenti al secondo (10 aperture e 10 chiusure) ci sarebbero un minimo di 3000 ore ovvero 125 giorni di

funzionamento, prima che si possa avere una buona probabilità che non si verifichi più il contatto. Anche se questo è il periodo *minimo* di vita e in realtà la maggior parte dei relé dura più a lungo, può rappresentare un problema, particolarmente nelle applicazioni di acquisizione dati remote. I relé reed sono accettabili in molte applicazioni di multiplexer analogici ma il loro impiego deve essere attentamente vagliato.

MULTIPLEXER ANALOGICI

Ci sono molti e differenti tipi di dispositivi a semiconduttore che possono essere usati per commutare un segnale, anche se quelli MOS hanno trovato un impiego più diffuso nelle commutazioni di segnali a basso livello o nelle applicazioni che richiedono multiplexer. La tecnologia di fabbricazione dei semiconduttori non è interessante ai fini della nostra discussione, ma sicuramente conoscerete i termini *complementary-metal-oxyde semiconductor* (CMOS) e *metal-oxyde semiconductor field effect* (MOSFET) che si riferiscono a due tecnologie usate per la fabbricazione di dispositivi a semiconduttore di commutazione e di multiplexer.

Alcuni dei vantaggi che rendono più interessanti i commutatori a semiconduttore rispetto ad altri tipi per un loro impiego nei multiplexer sono:

- Piccole dimensioni, formato standard dual-in-line
- Direttamente compatibili con logica TTL
- Decodificatori digitali per la selezione del canale interno
- Segnale di ingresso positivo e negativo (segnali bipolari)
- Alta velocità di commutazione
- Lunga durata, nessuna usura meccanica
- Bassa resistenza di contatto, meno di 100 Ohm
- Alta resistenza nello stato aperto, generalmente 10^9 Ohm

MULTIPLEXER ANALOGICI A SEMI-CONDUTTORI

CONSIDERAZIONI

I commutatori a semi-conduttore non sono dispositivi ideali e quindi comportano alcune limitazioni o restrizioni di impiego che devono essere tenute in considerazione prima del loro utilizzo nei circuiti di multiplexer. Quasi tutti i commutatori analogici che usano dispositivi a semi-conduttore per realizzare la funzione di commutazione richiedono due alimentazioni, generalmente + 15 Volt e − 15 Volt, e i segnali di ingresso non possono superare questi valori senza che avvenga il danneggiamento del dispositivo.

In un primo tempo essi erano soggetti al fenomeno chiamato *latch-up* che li portava a comportarsi come dei rettificatori controllati al silicio (SCR).Ciò significa che una volta chiusi per far passare il segnale essi non si aprivano fino a che il segnale di ingresso non avesse raggiunto 0 Volt. Dato che il segnale di ingresso non deve necessariamente raggiungere questo valore i commutatori rimanevano nello stato di chiusura (on) indefinitamente o finché non veniva tolta l'alimentazione dal sistema.

Molti dei dispositivi a semi-conduttore, in particolare quelli CMOS e MOSFET sono facilmente danneggiabili dalle cariche elettriche statiche che possono essere prodotte da manufatti sintetici, comerte ecc. A questo proposito i più recenti progetti vengono realizzati in modo da contenere dispositivi di protezione statica all'interno del circuito integrato.

Ci sono alcuni termini relativi ai multiplexer a semi-conduttore che ne definiscono meglio le caratteristiche:

CROSSTALK È la misura del valore di un segnale che è in ingresso ad un canale nello stato off, ma che appare all'uscita del multiplexer sovrapposto al segnale che interessa e che è attivato in quel momento. Questo fenomeno è in funzione della frequenza dei segnali che sono in ingresso al multiplexer in quanto i commutatori sono a causa della capacità, accoppiati all'interno del circuito integrato e, di conseguenza, alcuni dei segnali si disperdono attravero altri canali. Il crosstalk aumenta in genere con l'aumentare della frequenza dei segnali in ingresso al multiplexer ed è simile al problema feedthrough associato ai dispositivi S/H.

SETTLING TIME Il tempo di assestamento rappresenta il tempo necessario perché l'uscita del multiplexer si trovi all'interno di una certa percentuale di errore rispetto al segnale di ingresso una volta che il canale sia selezionato, cioé posto nello stato on. È definito come tempo di commutazione più il tempo di stabilizzazione dell'uscita analogica, o solo come tempo di assestamento dell'uscita analogica.

THROUGHPUT RATE Questa grandezza è la misura della massima velocità di commutazione da canale a canale che possa essere effettuata rimanendo all'interno di una percentuale di variazione accettabile, normalmente pari allo 0,01%.

LARGHEZZA DI BANDA Definisce la capacità del multiplexer a far passare un segnale di una particolare frequenza una volta che il canale sia on. La larghezza di banda è rappresentata dal punto –3dB ed equivale alla larghezza di banda per piccoli segnali dei dispositivi S/H.

TRANSISTORI DI COMMUTAZIONE Quando un multiplexer è commutato da un canale al successivo ed uno dei canali è aperto, avviene un transitorio di commutazione che comporta dei picchi di tensione che appaiono all'uscita del multiplexer e possono causare misure non accurate se l'uscita viene campionata digitalizzata o integrata in questo istante. Essi possono essere eliminati, qualora sia necessario, usando un dispositivo S/H posto tra l'uscita del multiplexer e l'ingresso del dispositvo di misura.

SEGNALI IN INGRESSO DEI MULTIPLEXER

Esiste una vasta gamma di sorgenti che possono fornire segnali da multiplexare e possono essere costituite da segnali provenienti da termocoppie a basso livello, o da trasduttori di pressione, ad alto livello, così come segnali D.C e A.C o ad alta e bassa frequenza. Tutti questi tipi di segnali possono essere multiplexati con successo, anche se possono richiedere alcuni condizionamenti prima e dopo l'operazione di multiplexer.

SEGNALI A BASSO LIVELLO Possono richiedere un'amplificazione prima che vengano posti in ingresso ai multiplexer, dal momento che i disturbi dovuti ai transitori possono essere sufficientemente grandi da causare errori significativi nell'uscita a basso livello del multiplexer. Se necessario, i segnali una volta multiplexati e amplificati possono essere nuovamente attenuati. Se l'amplificazione non può essere effettuata su tutto il campo dei valori in ingresso, è possibile utilizzare un filtro sull'uscita del multiplexer per eliminare i disturbi generati dai transistori di commutazione.

SEGNALI DIFFERENZIALI Ci sono spesso casi in cui il segnale che deve essere misurato non è costituito da una singola tensione riferita nel solito modo al potenziale di terra ma esista come differenza tra due tensioni; in questo caso viene appunto definito *segnale differenziale* e può avere un potenziale molto basso quando comparato alle reali tensioni presenti su due ingressi.

Per esempio si può immaginare di avere due tensioni, 5,37 Volt e 5,35 Volt e che la loro differenza di 0,02 Volt rappresenti il segnale che ci interessa. Questo segnale è effettivamente molto piccolo se paragonato ai due potenziali 5,37 e 5,35 Volt.

Della strumentazione con amplificatori differenziali in ingresso può essere utilizzata per rilevare la differenza producendo un solo valore come uscita da misurare. Quest'ultima può poi essere multiplexata e fornita ad un convertitore A/D per la digitalizzazione. Nel

caso in cui si debba misurare un numero elevato di segnali differenziali in questo modo, diventa costoso fornire per ogni coppia di linee differenziali in ingresso un amplificatore. Una soluzione valida a questo problema è quella di usare un multiplexer a *due poli*, che abbia due indipendenti posizioni di commutazione che permettano un'apertura e chiusura nello stesso istante. Il multiplexer a due poli viene usato per commutare entrambi le linee di un segnale differenziale direttamente dal multiplexer ad un comune amplificatore differenziale. In questo modo un singolo amplificatore può essere condiviso tra molte sorgenti differenziali

Un esempio di multiplexing differenziale o a doppio polo è riportato nella Figura 5-11, nella quale non è stato riportato nessun circuito di decodifica del canale dal momento che vengono usati solo due commutatori.

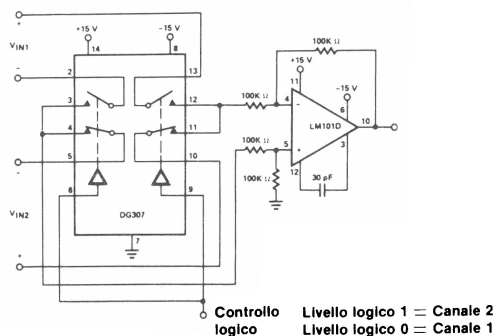



Figura 5-11. Circuito multiplexer a due canali con ingressi differenziali; un amplificatore differenziale viene usato per fornire una uscita ad una sola polarità. Per gentile concessione della Siliconix Inc.[4]

Il tipo di configurazione riportata nella Figura 5-11 può essere anche utilizzata quando entrambi i segnali differenziali e non, sono multiplexati e connessi allo stesso dispositivo di acquisizione che può essere un amplificatore, un convertitore A/D o un altro dispositivo. Gli ingressi a una sola polarità hanno il secondo segnale riferito o collegato ad un punto di massa analogica.

APPLICAZIONI DEL MULTIPLEXER

I commutatori analogici possono essere usati in quasi tutti i circuiti che richiedono la commutazione di una tensione, e le loro tipiche applicazioni sono nei convertitori D/A, negli amplificatori a guadagno programmabile, nei circuiti di filtro e negli integratori. Il nostro interesse principale si concentra sul loro impiego nei multiplexer analogici usati per commutare ingressi di segnali multipli ad un punto comune per l'amplificazione e la digitalizzazione. Esistono due tipi di dispositivi che prenderemo in considerazione: quelli senza decodificatori e quelli con decodificatori.

COMMUTATORI SENZA DECODIFICATORI Alcuni commutatori analogici come il TL182C della Texas Instruments e il 7510, 7511 e 7512 della Analog Devices hanno un controllo per gli ingressi di ogni commutatore e le loro configurazioni dei piedini sono riportati nella Figura 5-12.

Questo tipo di commutatori analogici richiedono un livello logico 1 o un livello logico 0 per l'attuazione di ciascuna linea e trovano impiego nelle applicazioni in cui più di una commutazione deve essere effettuata in un certo istante o in cui può essere necessario un controllo individuale del commutatore.

Un altro esempio di operazioni di multiplexer è riportato nella Figura 5-13, dove quattro termocoppie sono sottoposte a tecnica di multiplexer in modi diversi. Le due uscite differenziali del multiplexer sono collegate ad un unico amplificatore differenziale. Ogni termocoppia è commutata e collegata all'amplificatore applicando il livello logico appropriato all'ingresso CK della termocoppia selezionata. I commutatori sono controllati individualmente in modo da permettere, in questa configurazione, di collegare più di una termocoppia all'amplificatore nello stesso istante. In questo esempio si deve osservare

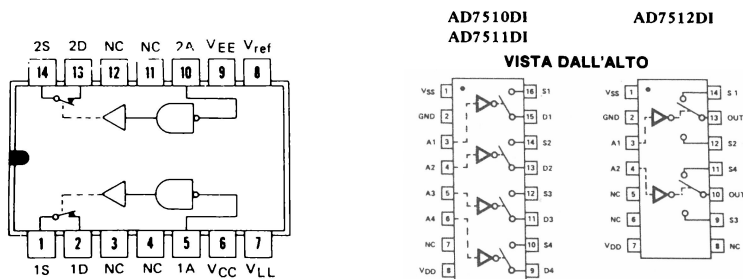



Figura 5-12. Configurazione dei piedini di circuiti integrati di multiplexer che non contengono logica di decodifica. Sono riportati il TL182C della Texas Instruments e la serie AD7510 della Analog Devices. Per gentile concessione delle case suindicate.

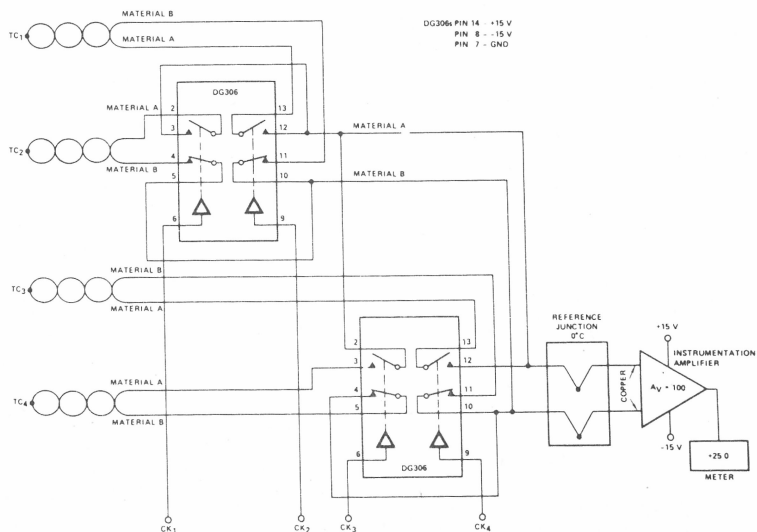



Figura 5-13. Un esempio di multiplexing a termocoppia in cui sono usate quattro termocoppie come fonti di segnale e un amplificatore di strumentazione per il recupero del segnale. Per gentile concessione della Siliconix, Inc.

che nonostante i segnali a basso livello della termocoppia siano multiplexati, non viene usata alcuna preamplificazione o altro condizionamento per i segnali.

COMMUTATORI CON DECODIFICATORI I commutatori utilizzati per multiplexare un segnale analogico sono generalmente più utili in quanto sono dotati di circuiti di decodifica che accettano un ingresso parallelo binario e conseguentemente attivano il contatto corrispondente al codice fornito di ingresso. Il codice binario può rappresentare un solo valore in un certo istante provocando l'attivazione di un solo commutatore per volta. Alcuni esempi di commutatori con decodifica sono riportati nella Figura 5-14 assieme alla tabella della verità di questi dispositivi.

Quando si usano multiplexer analogici con decodificatori in unico elemento, è ancora l'utente che deve fornire il codice opportuno per il canale richiesto. Molti decodificatori forniscono inoltre un ingresso di abilitazione che permette di espandere lo schema del multiplexer in modo da contenere un numero maggiore di canali selezionabili. Un tipico esempio è il multiplexer a 32 canali illustrato nella Figura 5-15, in cui è stato usato il multiplexer Siliconix DG506. Da notare l'uso dell'ingresso di abilitazione piedino 18, che viene usato per permettere la commutazione tra i due dispositivi multiplexer attivando uno e disattivando l'altro. Di fatto ciò non implica interventi concreti da parte dell'utente, dato che l'ingresso di abilitazione può essere collegato come se fosse semplicemente un altro ingresso di indirizzo. Usando gli ingressi di abilitazione e i decodificatori TTL addizionali, il multiplexer può essere espanso quasi all'infinito.

INTERFACCIAMENTO DI UN MULTIPLEXER

In molti casi di acquisizione dati, dove devono essere digitalizzati ingressi analogici, si usano multiplexer per poter suddividere tra gli ingressi un solo convertitore A/D. In questo tipo di sistema, è compito del progettista e del programmatore permettere al calcolatore di selezionare il canale corretto del multiplexer.

TABELLE DELLA VERITA'

AD7506					
A_3	A_2	A_1	A_0	E_N	"ON"
0	0	0	0	1	1
0	0	0	1	1	2
0	0	1	0	1	3
0	0	1	1	1	4
0	1	0	0	1	5
0	1	0	1	1	6
0	1	1	0	1	7
0	1	1	1	1	8
1	0	0	0	1	9
1	0	0	1	1	10
1	0	1	0	1	11
1	0	1	1	1	12
1	1	0	0	1	13
1	1	0	1	1	14
1	1	1	0	1	15
1	1	1	1	1	16
X	X	X	X	0	None

AD7507				
A_2	A_1	A_0	E_N	"ON"
0	0	0	1	1 & 9
0	0	1	1	2 & 10
0	1	0	1	3 & 11
0	1	1	1	4 & 12
1	0	0	1	5 & 13
1	0	1	1	6 & 14
1	1	0	1	7 & 15
1	1	1	1	8 & 16
X	X	X	0	None

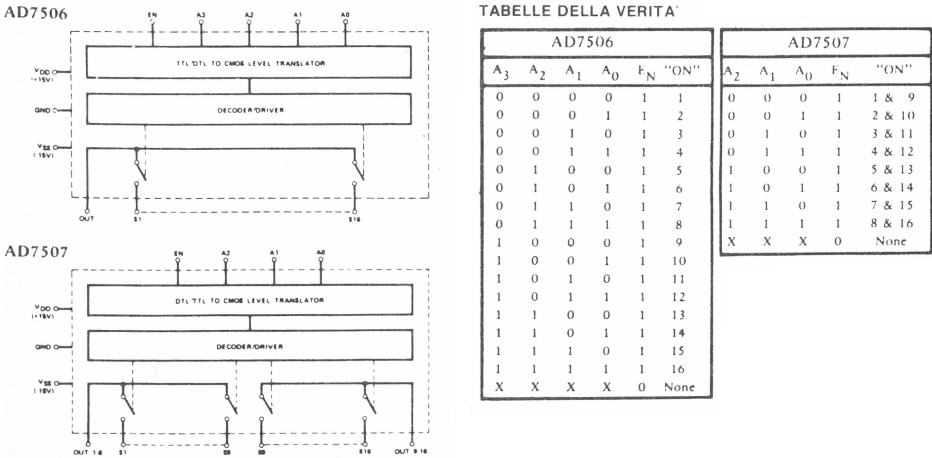



Figura 5-14. Schema a blocchi di alcuni tipi multiplexer con decodificatori in un unico elemento. Si tratta di due circuiti della Analog Devices e cioè l'AD7506 a sedici canali e l'AD7507 a otto canali.

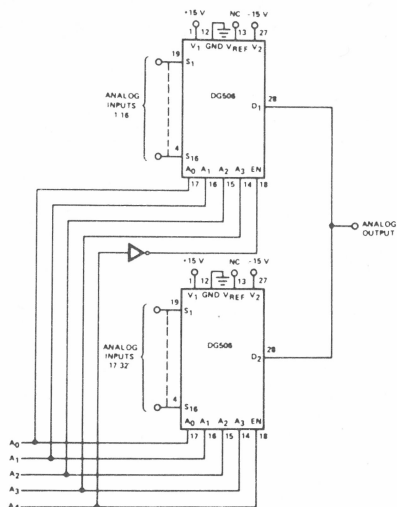



Figura 5-15. Schema di circuito di un multiplexer a 32 canali che utilizza i circuiti integrati del Siliconix DG506 e un invertitore.

I multiplexer con decodificatori in un solo elemento rappresentano probabilmente la scelta più saggia perché permettono di evitare l'aggiunta di nuova logica per la selezione del commutatore. Sarà necessario, comunque, avere una porta di uscita o un circuito di memorizzazione in modo che il codice del canale possa essere messo in uscita dal calcolatore e memorizzato per il tempo necessario. Le uscite del circuito di memorizzazione sono collegate semplicemente agli ingressi di indirizzo del canale sul dispositivo multiplexer. Se per l'interfaccia con il calcolatore deve essere usato lo schema illustrato nella Figura 5-15, le uscite del circuito di memorizzazione dovranno essere collegate agli ingressi A4 e A0 dell'indirizzo, i quali andranno collegati tramite il circuito di memorizzazione alle linee da D4 a D0 del bus del calcolatore. La parola binaria memorizzata alla porta di uscita dovrebbe controllare il canale selezionato.

Mentre una normale porta di uscita (con memoria) può fornire il necessario controllo del multiplexer, ci può essere in questo tipo di circuito un metodo di memorizzazione migliore o più corretto, quello, per esempio, fornito dal contatore binario, AVANTI-INDIETRO, programmabile SN74193 che presenta delle caratteristiche utili, che non si trovano nel circuito di memorizzazione, e sono:

- Memorizzazione parallela di un valore di conteggio
- Possibilità di incremento del valore di conteggio
- Capacità di decremento del valore di conteggio
- Capacità di azzerare il valore di conteggio

Con queste possibilità, possiamo ora caricare il numero di canali del multiplexer nel contatore SN74193 e incrementare o decrementare il valore di conteggio per commutare da un canale all'altro in sequenza, non solo, ma il valore di conteggio può essere cancellato per ripristinare il multiplexer sul canale zero. Nella Figura 5-16 troverete una tipica interfaccia per un multiplexer *programmabile* a 16 canali.

Va ricordato che il multiplexer è posto sotto controllo di programma e che possiamo scrivere i nostri programmi per sfruttare questa possibilità. Il programma per mettere in sequenza i 16 canali è piuttosto semplice; è sufficiente generare un comando per incrementare il valore di conteggio memorizzato nel contatore SN74193. In alcuni casi può essere necessario campionare un numero di canali analogici, i quali non possono però essere contigui o in sequenza. Per esempio occorre scandire o digitalizzare i segnali presenti nei canali 9, 3, 5, 2, 9, 3, 5, 2, ecc. Ecco nell'esempio che segue il programma da utilizzare:

```
MVIA    CARICA IL REGISTRO A CON IL NUMERO DI CANALE
011     011 = 9 DECIMALE
OUT     LO METTE IN USCITA VERSO IL MULTIPLEXER
351
CALL    ESEGUE UNA CONVERSIONE
ADC
0
MVIA    CARICA IL NUMERO DI CANALE SUCCESSIVO
003
OUT     LO METTE IN USCITA VERSO IL MULTIPLEXER
351
•
•
ETC
```

Esempio 5-1. Programma tipico usato per scandire una serie di canali preselezionati del multiplexer.

Nel programma illustrato nell'Esempio 5-1, tutti i canali da scandire sono indicati nel programma principale. Come vedrete sarà difficile cambiare i canali da scandire e il numero totale di canali da scandire su una base giornaliera, particolarmente se la sequenza di programma è memorizzata su PROM o ROM. Un approccio più generale è quello di usare una *tabella di scansione* che indichi ad una subroutine di scansione del multiplexer quali canali devono essere scanditi e digitalizzati. Una spiegazione dettagliata di questo tipo di programma è fornita nel Capitolo 6.

Nell'esempio che abbiamo esaminato e riguardante l'acquisizione di valori di temperature, il sistema poteva essere espanso in modo da comprendere altri sensori oltre ai sensori per l'umidità, i livelli dell'acqua, ecc. Ciascuno di questi sensori può essere collegato ad un multiplexer la cui uscita, a sua volta è collegata all'ingresso del convertitore A/D. Per alcuni sensori può essere necessario un adattamento dei segnali, a seconda dei livelli di corrente o di tensione in relazione al campo di variazione della tensione in ingresso del convertitore A/D. In caso di segnali che possono cambiare rapidamente, possono risultare particolarmente utili i moduli sample-and-hold. Inoltre, può essere richiesta, la visualizzazione multipla di gruppi di dati relativi a ciascun sensore, per cui è necessario un programma di controllo che rilevi delle condizioni specifiche per ogni sensore.

Si auspica che sviluppi futuri nel campo dei convertitori A/D conducano entro breve tempo, alla realizzazione di piccoli sistemi di acquisizione dati, già assemblati, contenente multiplexer analogici, circuiti sample-and-hold e convertitori A/D veloci e precisi. I prezzi relativi sia alle "scatole nere" di acquisizione dati che ai convertitori monolitici a un solo chip e in un unico elemento continuano a diminuire, facendoci sperare nell'avvento di convertitori meno costosi che permettano l'impiego di un convertitore A/D per ogni sensore.

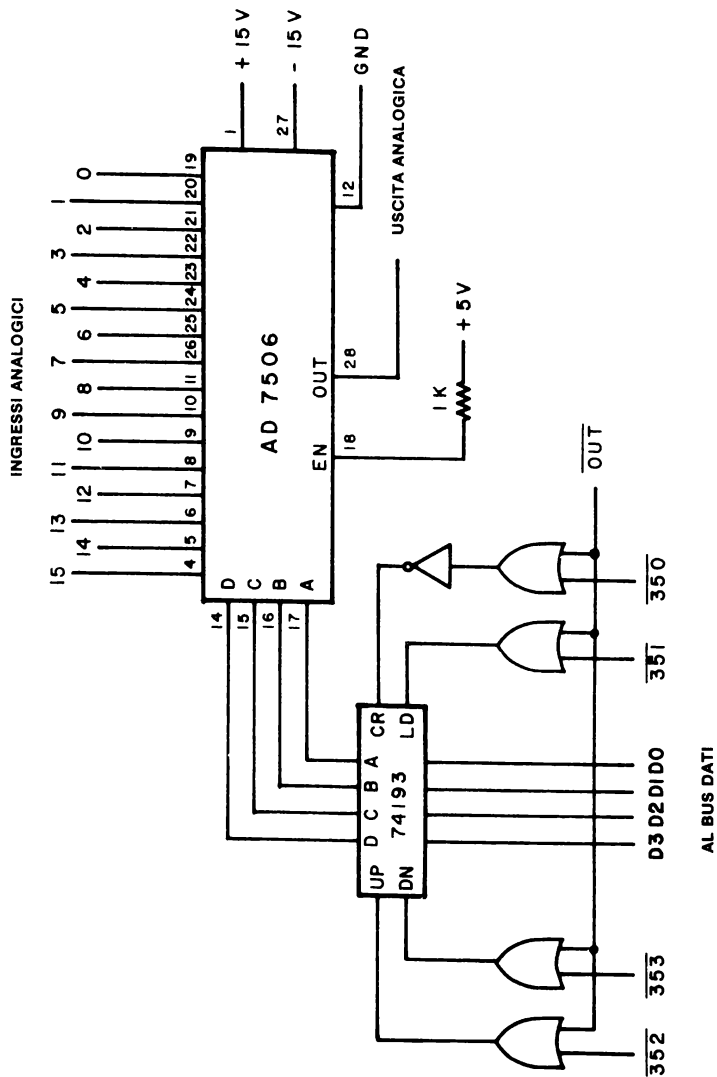



Figura 5-16. Multiplexer analogico con controllo a microcalcolatore. I numeri dei canali possono essere selezionati e controllati mediante comandi di programma.

SISTEMI DI MULTIPLEXER E SAMPLE-AND-HOLD

I dispositivi multiplexer e sample-and-hold sono particolarmente utili quando vengono impiegati assieme ai sistemi di acquisizione di dati. In questi casi sono possibili due configurazioni, una in cui, tra un multiplexer analogico e un convertiore A/D viene usato un unico dispositivo sample-and-hold e una in cui ogni canale ha il suo dispositivo sample-and-hold. Nella Figura 5-17 è illustrato un esempio della prima configurazione.

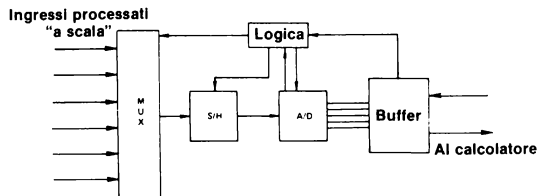



Figura 5-17 Un tipico sistema di acquisizione dati che utilizza un singolo modulo sample-and-hold e un multiplexer.

Il circuito illustrato nella Figura 5-17 rappresenta una soluzione economica di un problema di acquisizione dati a più canali e può essere facilmente espanso aggiungendo più multiplexer. Il sistema funziona commutando sul canale attivo che interessa, campionandolo e poi digitalizzandolo. Il procedimento può, comunque, richiedere decine di microsecondi di ritardo tra le misure effettive. Tale ritardo è dovuto al tempo di commutazione del multiplexer, al tempo di acquisizione e il tempo di setting (assestamento) del circuito sample-and-hold e al tempo di conversione necessario al convertitore A/D. Il tempo tra i diversi campionamenti può essere leggermente ridotto commutando il multiplexer sul canale successivamente interessato, dopo che il valore è stato acquisito e mantenuto dal circuito sample-and-hold.

In un sistema come questo, gli ingressi possono essere digitalizzati in sequenza o casualmente. Il modo di funzionamento casuale viene utilizzato in applicazioni in cui è necessario acquisire o digitalizzare dati provenienti da canali a velocità diverse.

Il secondo sistema di acquisizione dati, invece, utilizza un dispositivo sample-and-hold per ogni ingresso analogico. Questo tipo di configurazione è più costoso di quello illustrato nella Figura 5-17, ma si rivela particolarmente utile in quelle applicazioni che richiedono misure simultanee di un certo numero di segnali analogici, come nel caso, per esempio, dell'analisi di fenomeni transitori o di misure legate alla frequenza prese durante prove di vibrazione.

In questo modo di funzionamento, tutti i dispositivi sample-and-hold sono commutati dal modo sample a quello hold nello stesso istante. I segnali analogici vengono così conservati fino a quando non vengono digitalizzati in sequenza, usando di nuovo un unico convertitore A/D con un multiplexer analogico per commutare i numerosi ingressi analogici verso il convertitore. Un tipico esempio è quello illustrato nella Figura 5-18.

Se la digitalizzazione richiede un certo tempo, piuttosto lungo, è necessario usare dispositivi sample-and-hold a bassa dispersione. La velocità di dispersione deve essere tale che l'uscita dell'ultimo canale da digitalizzare non cambi in modo tale da influenzare i valori legati al suo sensore. Se, per esempio, bisogna misurare 10 canali in un sistema a campionatura simultanea e se i canali sono posti in una sequenza di 20 microsecondi per canale, ci vorranno 180 microsecondi prima che inizi la digitalizzazione del valore analogico all'ultimo canale. Se si sta usando un convertitore A/D a 10 bit, la curva di dispersione dovrà essere tale che l'uscita dell'ultimo dispositivo sample-and-hold sia tale

che la variazione dell'uscita rimanga inferiore a una parte su 1024 per almeno 180 microsecondi.

In un caso come questo, in cui sono necessarie alte velocità di acquisizione e basse dispersioni, potrebbe risultare utile un sistema sample-and-hold in cascata, cioé simile a quello già esaminato in precendenza, e che abbiamo illustrato nella Figura 5-9. Come vedremo nel prossimo capitolo, il singolo dispositivo sample-and-hold è la soluzione più frequentemente usata per problemi di acquisizione dati.

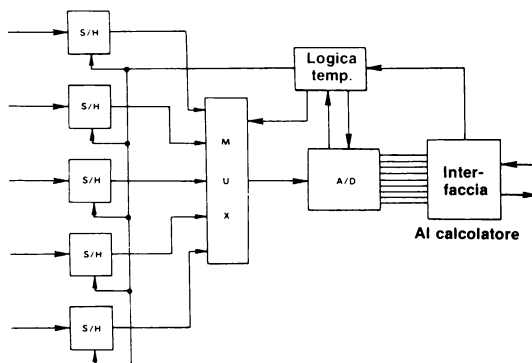



Figura 5-18. Un tipico sistema di campionatura simultanea a multiplexer nel quale è stato usato, con ogni canale di ingresso un dispositivo sample-and-hold.

ELABORAZIONE DI SEGNALI

La magggior parte dei dettagli sull'elaborazione di segnali vanno oltre lo scopo di questo libro, che si propone di esaminare i problemi relativi all'interfacciamento dei convertitori e le implicazioni software relative, mentre ciò che è importante è che risulti chiaro il significato di alcuni termini relativi all'elaborazione dei segnali. Vi forniremo inoltre alcuni elementi di consultazione di cui potreste aver bisogno nel caso in cui l'elaborazione dei segnali risulti rivestire una parte importante nella vostra specifica applicazione.

Nel corso di questo volume abbiamo dato per scontato la vostra capacità di emettere segnali limpidi e privi di disturbi da digitalizzare mediante convertitore A/D, o che siate in grado di usare la tensione o la corrente messa in uscita dal convertitore D/A. Si tratta comunque di condizioni ideali e può certamente succedere che la vostra applicazione richieda il ricorso ad un filtro per ridurre i rumori, più ampi campi dinamici per segnali che variano, come ordine di grandezza, di parecchie decadi o ancora a conversioni in cui è importante il rapporto tra un segnale e un altro e non i valori effettivi dei due segnali.

RUMORE Il rumore si può presentare sotto diversa forma e può essere ridotto mediante filtri, integrazione, correlazione di segnali ecc. oppure può essere accettato così com'è elaborato. In ogni caso, la causa di un disturbo è molto complessa ed è stata trattata meglio da altri. A questo proposito vi rimandiamo al volume *Instrumentation for Scientists Series, Module 4 Optimization of Electronic Measurements* e al libro *The Design of Active Filters*, che fanno parte della bibliografia di questo capitolo

CAMPO DINAMICO Il campo dinamico, o il numero di decadi oltre le quali il segnale può variare, può essere maggiore di quanto non possa essere trattato, con precisione, da molti convertitori. Per fornire la tensione di fondo scala necessaria ad un convertitore A/D si possono usare amplificatori con guadagno programmabile, con un

guadagno, cioé, che può essere programmato se necessario, a controllo di programma oppure può essere definito in modo automatico come nel caso di alcuni pannelli di misura digitali. Si possono usare anche amplificatori con una relativa risposta logaritmica per mettere in uscita, diciamo, un volt per decade di tensione di ingresso. Così, un cambiamento da 0,01 a 0,1 volt all'ingresso genera un cambiamento di un volt nell'uscita dell'amplificatore logaritmico, come lo genererebbe il cambio da 0,1 a 1,0 volt.

CONVERSIONI È spesso necessario misurare il rapporto tra un segnale ed un altro, piuttosto che i valori effettivi di entrambi. Ciò può essere realizzato digitalizzando ciascun valore e poi calcolando il rapporto o il valore matematico in software o usando un convertitore A/D per misure di rapporti, che accetta due ingressi e mette in uscita un valore che rappresenta il rapporto tra i due.

BIBLIOGRAFIA

1. Sheingold, D. H., ed., *Analog-Digital Conversion Notes,* Analog Devices, Inc., Norwood, MA 02062, 1977.
2. Bruck, D.B., *Data Conversion Handbook,* Hybrid Systems Corporation, Burlington, MA 01803, 1974.
3. National Semiconductor Corporation, *LF-389 Data Sheet,* National Semiconductor Corporation, Santa Clara, CA 95051, 1976.
4. Siliconix, Inc., *Analog Switches and Their Applications,* Siliconix, Inc., Santa Clara, CA 95054, 1976.
5. Malmstadt, H.V., *et al., Instrumentation for Scientists Series, Module 4, Optimization of Electronic Measurements,* W.A. Benjamin, Inc., Menlo Park, CA, 1974.
6. Berlin, H.M., *The Design of Active Filters, with Experiments,* E&L Instruments, Inc., Derby, CT 06418, 1977.

CAPITOLO 6

... DEI BIT, SCHEDE E SCATOLE NERE

INTRODUZIONE

In questi ultimi anni, i costruttori di moduli analogici hanno reso più semplice il compito di interfacciare convertitori analogici/digitali, incorporando all'interno dei moduli di conversione dispositivi dedicati all'interfacciamento come le uscite "three-state" o gli ingressi con memorizzazione. Con l'introduzione di moduli a microcalcolatore da parte di società come la "Control Logic" e la "Pro-Log" e di calcolatori su una singola scheda, come quelli disponibili presso l'Intel, la Motorola e la National Semiconductor. I costruttori di moduli hanno iniziato a fornire interfacce analogiche complete e pronte all'uso e che risultano compatibili con i vari segnali del bus. Queste schede o cartoline costituiscono un semplice modo per aggiungere possibilità di conversioni digitali e analogiche a piccoli sistemi con i calcolatori.

LE SCATOLE NERE - MODULI PER ACQUISIZIONE DATI

I dispositivi modulari monolitici o compositi per l'acquisizione di dati in genere forniscono una completa interfaccia tra il conduttore analogico/digitale e i sistemi a microcalcolatore o a minicalcolatore. Esclusi i casi nei quali particolari esigenze non sono facilmente risolvibili, è meno costoso acquistare un modulo per acquisizione dati, piuttosto che tentare di costruirne uno in proprio.

Tali moduli attualmente costano intorno ai 300$ ciascuno, sebbene la recente tendenza indichi che i moduli diventeranno sempre più piccoli e meno costosi mano a mano che i costruttori cominceranno ad integrare la circuiteria necessaria all'interno di un singclo circuito integrato. I moduli per acquisizione dati attualmente disponibili hanno le seguenti caratteristiche:

- Ingressi analogici singoli a sedici canali o differenziali a otto canali
- Controllo del multiplexer completo per selezione di canale sequenziale o casuale
- Amplificatore differenziale e dispositivo di sample-and-hold interni
- Convertitore A/D binario a dodici bit
- Logica interna per temporizzazione e controllo
- Contenitore da 7,5 x 12,5 cm
- Velocità di conversione fino a 50 kHz

Tutta la circuitèria di controllo necessaria è contenuta nel modulo, compreso il controllo del circuito sample-and-hold e del convertitore A/D. L'utente deve fornire i collegamenti ai dati del microcalcolatore e ai segnali del bus di controllo.

I moduli di acquisizione dati vengono forniti da molte case produttrici di componenti.

Fra cui i seguenti:

ADAC Corporation, Woburn, MA 01801

ADAM-12	16-Channel, 12-bit A/D Converter

Analog Devices, Inc., Norwood, MA 02062

DAS1128	16-Channel, 12-bit A/D Converter

Analogic Corporation, Wakefield, MA 01880

MP6812	16-Channel, 12-bit A/D Converter

Burr-Brown Research Corp. Tucson, AZ 85734

SDM853	16-Channel, 12-bit A/D Converter
MP-20	16-Channel, 8-bit A/D Converter

Data Translation, Inc., Framingham, MA 01701

DT5701	16-Channel, 12-bit A/D Converters
DT820	8-Channel, 8-bit A/D Converter

Datel Systems, Inc., Canton, MA 02021

MDAS-16	16-Channel, 12-bit A/D Converter
MDAS-8D	8-Channel differential, 12-bit A/D

Nella Figura 6-1 è illustrato un tipico modulo di acquisizione dati, in cui dovreste individuare facilmente il multiplexer analogico, il sample-and-hold e il convertitore analogico/digitale a 12 bit. Noterete inoltre che, per fornire l'indirizzo a quattro bit al multiplexer, è stato usato un contatore programmabile. È stato aggiunto inoltre un amplificatore differenziale in modo che questo modulo particolare possa essere usato sia in modo single-end che differenziale.

Ci sono alcune caratteristiche importanti dei moduli di acquisizione dati che dovrebbero essere presi in considerazione con cura prima di essere usati in un'interfaccia. Ecco qui elencata una serie di domande per aiutarvi nella scelta:

- I dispositivi interni sono tutti collegati e pronti per l'uso?

  Alcuni moduli possono avere multiplexer, sample-and-hold e convertitori A/D non adatti. In questo caso, potete collegare i dispositivi se necessario, tuttavia è difficile usare il modulo in modo particolare se questo è il primo tentativo di interfacciamento. In ogni caso la maggior parte delle case produttrici forniscono informazioni dettagliate che riguardano sia la messa a punto che l'uso dei loro dispositivi.

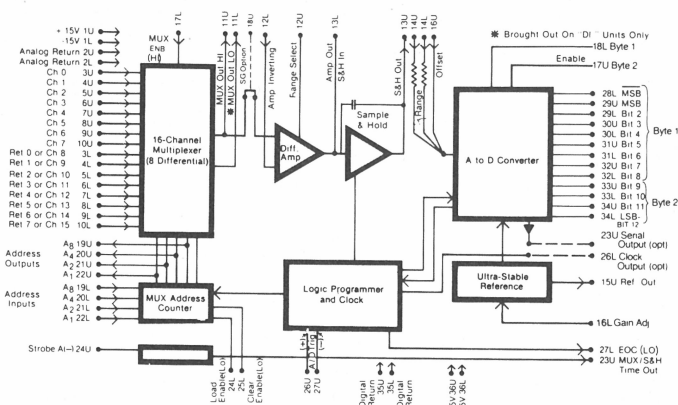



Figura 6-1. Un tipico modulo di acquisizione dati a 16 canali con un convertitore A/D a 12 bit e logica di controllo. Si tratta di un modulo DATAXTM della Data Translation, Inc.

- Gli ingressi e le uscite sono compatibili con i segnali forniti e richiesti dal microcalcolatore?

  Alcuni dei moduli disponibili hanno linee di uscita di dati paralleli three-state che sono compatibili con la maggior parte dei bus dati del microcalcolatore. È il caso del modulo DATAX illustrato nella Figura 6-1, il quale è fornito inoltre di linee specifiche per le funzioni di controllo; ciò permette il caricamento dei dati a 12 bit provenienti dal convertitore A/D in un byte a otto bit e in un byte a quattro bit.

  I moduli come il DAS1128 della Analog Devices e il DT820 (Traduttore di Dati) non hanno uscite three-state. Ciò rende leggermente più complesso l'interfacciamento; dato che è necessario fornire buffer three-state che permettono ai moduli di acquisire dati di essere compatibili con il bus three-state del microcalcolatore. I dispositivi come l'8212 e il DM8095 (SN74365) non sono costosi ma sono necessari decodificatori e altri elementi di circuito aumentando la circuiteria totale di interfaccia.

- L'alimentazione è adatta al sistema a microcalcolatore?

  I distributori di microcalcolatori e microprocessori danno una grande importanza all'uso di un'unica tensione di alimentazione, di solito + 5 volt, per alimentare un piccolo sistema a microcalcolatore. Se anche il vostro è un piccolo sistema dovrete aggiungere i + 15 volt e i − 15 volt necessari per i moduli di acquisizione dati. Quasi tutti i moduli richiedono + 5 volt, ma per evitare disturbi potete aggiungere + 5 volt separati, necessari per il modulo. La nostra esperienza ci ha dimostrato che l'alimentazione open-frame richiede spesso un ulteriore filtro se deve essere impiegata con moduli di acquisizione dati di grande precisione.

- È possibile aumentare il numero di canali analogici? Gli ingressi possono essere usati nel modo single-end o nel modo differenziale senza dover ricorrere ad un modulo diverso?

  Molti moduli possono essere espansi in modo da contenere altri canali analogici. Questo di solito richiede un altro modulo multiplexer. Va ricordato che 64 è il numero massimo di canali che possono essere controllati da un modulo di acquisizione dati.

La maggior parte dei moduli possono essere usati sia nel modo single-end che nel modo differenziale senza dover ricorrere ad un altro dispositivo.
Per alcuni moduli tuttavia, sarà necessaria nuova circuiteria nel caso in cui voleste allargare il numero di canali analogici o usare ingressi differenziali invece di ingressi single-end.

INTERFACCIAMENTO DI UN MODULO DI ACQUISIZIONE DATI

L'interfacciamento di un modulo di acquisizione dati od un microcalcolatore non è difficile in modo particolare quando, per questo scopo, viene impiegato un modulo con uscite di dati three-state. Infatti i moduli senza uscite three-state richiedono buffer three-state addizionali da collegare al bus dati del microcalcolatore. Nella Figura 6-2 è illustrata una tipica interfaccia nella quale è stato usato un modulo ADAM-12 della ADAC Corporation.

L'interfaccia dell'ADAM-12 fornisce la decodifica degli indirizzi per un I/O tramite accumulatore benché si possa usare anche un I/O mappato in memoria. Per semplificare la codifica degli indirizzi illustrata nella Figura 6-2 abbiamo scelto il DM8160, un circuito integrato comparatore binario a sei bit.

Qui di seguito sono elencati i quattro segnali di controllo e la rispettiva funzione:

OUT	300	Carica i quattro bit meno significativi del registro A dell'8080 (accumulatore) nel dispositivo di selezione del canale.
OUT	301	Fa partire il processo di conversione nell'ADAM-12.
IN	300	Legge gli otto bit meno significativi dei dati del convertitore A/D.
IN	301	Legge i quattro bit più significativi dei dati del convertitore A/D e il flag di fine/conversione (nella posizione D7).

Come potete osservare, sono necessari oltre al modulo ADAM-12, solo quattro circuiti integrati addizionali, che sono: il decodificatore SN7442, l'invertitore SN7404, il buffer three-state, SN74126 e il comparatore DM8160

PROGRAMMA DI ACQUISIZIONE DATI

Il programma per il controllo del modulo ADAM-12 è molto simile al programma di controllo di un convertitore A/D a 10 bit. La differenza più importante è che prima dell'inizio della conversione è necessario fornire al modulo un indirizzo per il multiplexer da 0 a 15. Quando si deve usare uno solo dei 16 canali all'inizio del programma si può caricare nel multiplexer l'indirizzo del canale scelto. Non ci sarà nemmeno necessità di modificarlo o aggiornarlo non appena il programma entra in fase di esecuzione.

È interessante notare che i dati messi in uscita dal modulo ADAM-12 è di fatto il contrario di quello previsto. È evidente che in qualche parte del sistema ha luogo un'inversione che potrebbe essere corretta usando una circuiteria addizionale per eseguire l'inversione dei dati, anche se la stessa funzione può essere rivestita da un comando di programma. In questo caso useremo l'istruzione di complemento del registro A.

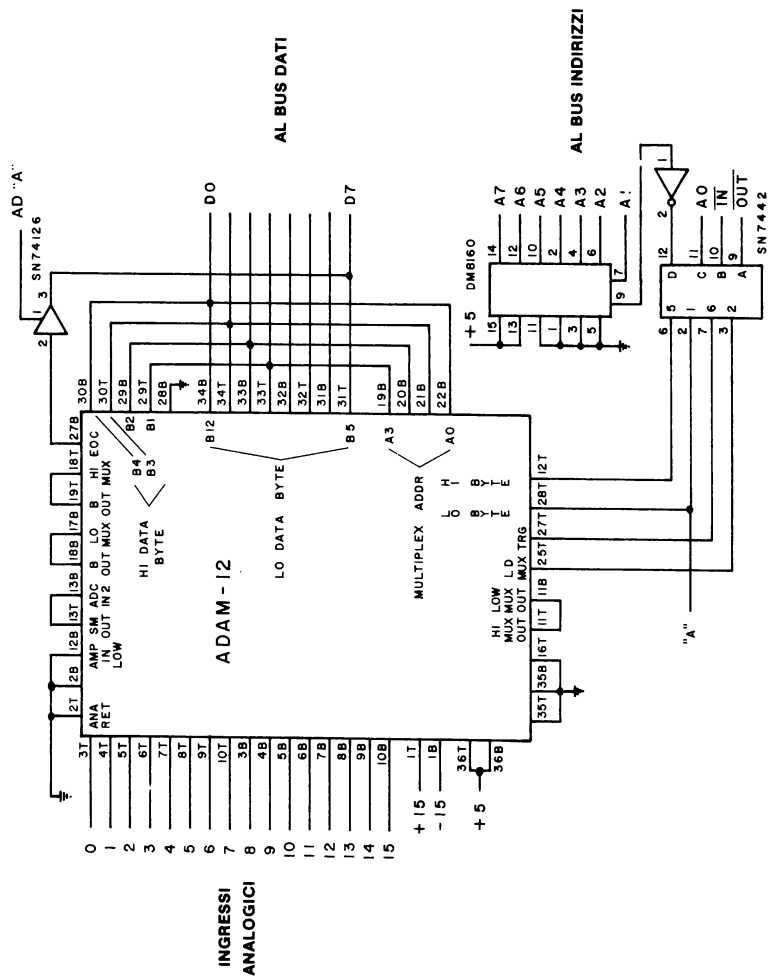



Figura 6-2. Ecco un esempio di interfaccia di un modulo di acquisizione dati ADAM-12. Gli unici collegamenti con il microcalcolatore sono il bus dati a otto bit, la parte bassa dell'indirizzo a otto bit e i due segnali di controllo $\overline{IN}$ e $\overline{OUT}$.

```
                    ESEMPIO 6-1
                    PROGRAMMA DI ACQUISIZIONE DATI DI UN
                    MODULO ADAM-12 DELLA ADAC CORPORATION

                    *030 000
030 000 061  START,   LXISP    CARICA IL PUNTATORE DI STACK
030 001 377           377
030 002 030           030
030 003 303           JMP      LA PRIMA VOLTA SALTA OLTRE LA ROUTINE DI
                               RITARDO
030 004 011           LOOP + 3
030 005 030           0
030 006 315  LOOP,    CALL     ESEGUE UN RITARDO
030 007 250           DELAY
030 010 030           0
030 011 016           MVIC     PREDISPONE UN CONTATORE
030 012 020           020      020 = 16 DECIMALE
030 013 021           LXID     PREDISPONE L'INDIRIZZO DELL'AREA DI
                               MEMORIZZAZIONE
030 014 202           STORE
030 015 030           0
030 016 052           LHLD     METTE LE PAROLE DI STATO IN H & L
030 017 200           STATUS
030 020 030           0
030 021 051  NEXT,    DADH     ESEGUE UNA ROTAZIONE DI 16 BIT
030 022 322           JNC
030 023 031           OVER
030 024 030           0
030 025 043           INXH     SE C'È UN RIPORTO, PONE UN 1 NEL BIT
                               MENO SIGNIFICATIVO
030 026 303           JMP      SE LO STATO È 1, LEGGE I DATI
030 027 042           CONVRT
030 030 030           0
030 031 023  OVER,    INXD     SE NON È 1 SPOSTA IL PUNTATORE DI
                               INDIRIZZO
030 032 023           INXD     PER LA MISURA SUCCESSIVA
030 033 015           DCRC     TUTTE LE MISURE ESEGUITE?
030 034 302           JNZ      NO, ESEGUE LA SUCCESSIVA
030 035 021           NEXT
030 036 030           0
030 037 303           JMP      SI', REINIZIALIZZA
030 040 006           LOOP
030 041 030           0


                    PROGRAMMA DI CONVERSIONE ADC PER IL CONTROLLO
                    MODULO ADAM-12

030 042 015  CONVRT, DCRC      DECREMENTA IL CONTATORE
030 043 171           MOVAC    LO SPOSTA IN A
030 044 323           OUT      LO METTE IN USCITA COME INDIRIZZO PER NPX
030 045 300           300
030 046 323           OUT      INIZIA UNA CONVERSIONE
030 047 301           301
030 050 333  EOC,     IN       LEGGE LO STATO ADC E I QUATTRO BIT PIU'
                               SIGNIFICATIVI
030 051 301           301
030 052 267           ORAA     ATTIVA IL FLAG
030 053 372           JM       NON ESEGUITO. RIPETE IL TEST.
030 054 050           EOC
```

```
030 055 030          0
030 056 057          CMA        ESEGUITO, COMPLEMENTA I DATI
030 057 346          ANI        MASCHERA I BIT NON USATI
030 060 017          017
030 061 107          MOVBA      MEMORIZZA I DATI IN B
030 062 333          IN         LEGGE GLI OTTO BIT MENO SIGNIFICATIVI
030 063 300          300
030 064 057          CMA
030 065 022          STAXD      MEMORIZZA I DATI
030 066 023          INXD
030 067 170          MOVAB
030 070 022          STAXD
030 071 023          INXD
030 072 171          MOVAC      CONTROLLA IL CONTATORE
030 073 247          ANAA       ATTIVA IL FLAG
030 074 312          JZ
030 075 006          LOOP
030 076 030          0
030 077 303          JMP
030 100 021          NEXT
030 101 030          0

                     AREA DI MEMORIZZAZIONE DEI DATI

                     *030 200
030 200 000 STATUS,  0          BIT DI STATO PER I CANALI 7-0
030 201 000          0          BIT DI STATO PER I CANALI 15/8
030 202 000 STORE,   0          L'AREA DI MEMORIZZAZIONE DATI INIZIA QUI
                                E UTILIZZA 32 LOCAZIONI

                     SUBROUTINE DI RITARDO STANDARD

                     *030 250
030 250 365 DELAY,   PUSHPSW
030 251 325          PUSHD
030 252 021          LXID
030 253 000          000        BYTE DI TEMPORIZZAZIONE
030 254 110          110
030 255 033 DEC,     DCXD
030 256 172          MOVAD
030 257 263          ORAE
030 260 302          JNZ
030 261 255          DEC
030 262 030          0
030 263 321          POPD
030 264 361          POPPSW
030 265 311          RET
```

Il programma dell'Esempio 6-1 illustra il modo in cui si può usare un microcalcolatore per acquisire e memorizzare dati provenienti da un numero massimo di 16 canali. Ogni canale analogico può essere attivo (ON) o inattivo (OFF), a seconda delle diverse applicazioni. I canali on e off sono rappresentati da una parola di stato a 16 bit memorizzata in due byte consecutivi di memoria a lettura/scrittura che può essere fissata prima di usare il programma di acquisizione dati.
Il metodo più efficace di stabilire i canali attivi e quelli inattivi consiste nell'usare comandi introdotti da commutatori o da telescriventi; mentre per predisporre la parola di stato a 16 bit verranno usate istruzioni del programma, che però non sono contenute nell'Esempio 6-1.

I dati a 12 bit provenienti dal convertitore A/D vengono memorizzati in due locazioni consecutive di memoria a lettura/scrittura associate ad un particolare canale. Se il canale è inattivo, non viene memorizzato nessun dato nuovo. I canali saranno scanditi in sequenza dal canale 15 fino al canale 0 e poi di nuovo al canale 15.

Per far si che i canali atti possano essere scanditi con un intervallo tra un'acquisizione e l'altra, viene fornita una chiamata alla subroutine di ritardo, DELAY. Il tempo reale di ritardo viene determinato dai due byte di temporizzazione annotati nella subroutine di DELAY, i quali possono essere modificati a seconda del particolare uso. Se per esempio, il ritardo viene stabilito di un secondo, tutti i canali attivi verranno campionati velocemente, quindi il calcolatore aspetterà per un secondo prima di riprendere la campionatura.

Per indicare il momento in cui le conversioni e le scansioni devono essere eseguite, si può usare anche un timer esterno basato su un interrupt o un timer di flag, che però mancano dal nostro esempio 6-1.

ALTRI MODULI DI ACQUISIZIONE DATI

Un altro tipo di modulo di acquisizione dati è il dispositivo MP-20 della Burr-Brown Research Corp., un piccolo modulo studiato appositamente per essere interfacciato con i chip del microprocessore tipo 8080 come l'8080A, l'8085 e lo Z-80. L'MP-20 è più piccolo della maggior parte dei moduli "scatola nera" fin qui menzionati, e presenta alcune caratteristiche che vale la pena di esaminare:

- L'MP-20 possiede una logica di decodifica interna per cui può essere collegato direttamente al bus indirizzi dell'8080. Gli indirizzi effettivi usati con il modulo vengono selezionati dall'utente applicando i livelli logici zero o uno agli undici piedini comparatori per la scelta dell'indirizzo.

- La tecnica di I/O con mappaggio della memoria viene usata per controllare il dispositivo e permette di utilizzare tutte le istruzioni con riferimento alla memoria. L'MP-20 può essere usato anche con I/O tramite accumulatore qualora fosse necessario.

- Il convertitore A/D possiede otto bit di risoluzione. Il circuito sample-and-hold non è incorporato nel dispositivo. Ad ogni canale dovrebbero essere aggiunti dei dispositivi sample-and-hold e ciò richiederebbe nuova circuiteria di controllo.

- Lo stato del convertitore è indicato da un'uscita READY, che può essere collegata direttamente all'ingresso READY dell'8080, ponendo così l'8080 nello stato WAIT finché la conversione non è stata completata. Ciò significa che l'8080 non può eseguire nessun altro passo di programma mentre è in attesa della fine della conversione. L'uscita READY dell'MP-20 può inoltre essere usata come flag per una gestione a controllo di programma o a interrupt. Va ricordato comunque che non è three-state.

Nella Figura 6-3 potete osservare lo schema a blocchi dello MP-20, uno strumento che noi pensiamo troverà posto in una vasta gamma di applicazioni in molti sistemi a microcalcolatore dove sono sufficienti otto bit di risoluzione e non è necessario il circuito sample-and-hold. Per maggiori informazioni la Burr-Brown è in grado di mettere a disposizione una documentazione particolarmente dettagliata. Per i microcalcolatori del tipo 6800 è disponibile un dispositivo equivalente, cioé l'MP-21.

SCHEDE: INGRESSI E USCITE ANALOGICI COMPATIBILI A SPINA

Molti dei produttori di convertitori analogici e di microcalcolatori interfacciano ormai le schede di ingresso e uscita analogiche che sono *compatibili* a livello di bus con molti dei

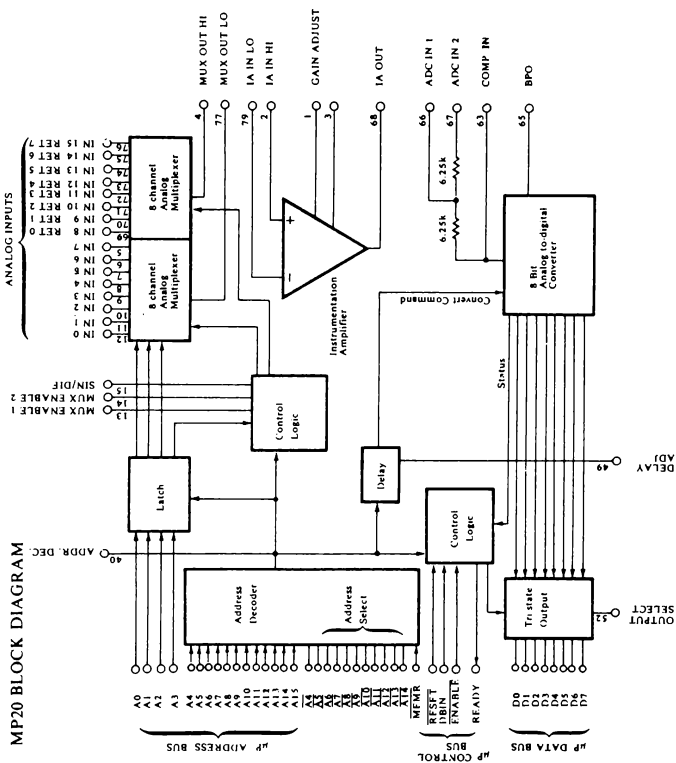



Figura 6-3. *Schema a blocchi di un modulo di acquisizione dati a 16 canali MP-20. Va notato che il multiplexer, l'amplificatore e il convertitore A/D non sono collegati l'uno con l'altro. Questi collegamenti, è sottointeso, devono essere forniti dall'utente.*

sistemi a microcalcolatore esistenti. Questo significa che l'interfacciamento analogico/digitale è spesso ridotto alla selezione, acquisto ed inserimento della scheda di I/O analogica adatta allo scopo.

Queste interfacce con il convertitore sono disponibili sia nella configurazione con mappaggio della memoria che nella configurazione di I/O tramite accumulatore. Alcune sono disponibili in entrambi i tipi la cui scelta spetta all'utente che può utilizzare diverse configurazioni realizzabili tramite opportuni collegamenti con piccoli ponticelli.

ADAC Corporation, Woburn, MA 01801

735 Series	16-Channel, 12-bit A/D, 2 12-bit D/A converters and an on-board clock

Analog Devices, Inc., Norwood, MA 02062

RTI-1200	16-Channel, 12-bit A/D, 2 12-bit D/A converters, 2708 PROM socket, real-time clock and control outputs

Burr-Brown Research Corporation, Tucson, AZ 85734

MP8616	16-Channel, 12-bit A/D Converter
MP8616A0	MP8616 with an 8-bit D/A

Data Translation, Inc., Framingham, MA 01701

DT1571	16-Channel, 12-bit A/D, 2 12-bit D/A converters

Datel Systems, Inc., Canton, MA 02021

ST-80016D	16-Channel (differential), 12-bit A/D converter
ST-800DA4	4 12-bit D/A converter

Molte delle schede analogiche di I/O hanno bisogno di un programma più esteso per funzionare in modo efficace, e come certamente saprete i programmi occupano una grossa fetta del tempo richiesto dall'interfacciamento. La cosa è valida anche nel caso di interfacce fornite nella forma compatibile a livello di bus. La maggior parte dei manuali dell'utente forniti assieme alle schde di I/O analogiche, mancano in modo sensibile di questa parte di programma. Alcuni produttori forniscono routine di inizializzazione e di collaudo ma pochi forniscono esempi di programmi che potrebbero essere utili nella realizzazione di interfacce.

Un'eccezione è costituita dal manuale dell'utente relativo all'interfaccia RTI-1200 della Analog Devices, che contiene più di una dozzina di esempi di programmi utili sia in fase di inizializzazione e calibratura che nel corso dell'effettiva applicazione. Il manuale di per sé conta più di 50 pagine.

Le schede analogiche di I/O disponibili rapppresentano un modo facile di interfacciare segnali analogici ai sistemi a microcalcolatore, semplificando il progetto dell'hardware, tuttavia il loro impiego deve essere valutato attentamente. Infatti possono essere troppo complesse e troppo costose per il problema che si sta affrontando, riducendo così i vantaggi complessivi del lavoro. Anche il programma per il funzionamento della scheda può risultare complesso, finendo con il richiedere più tempo di quello che sarebbe necessario per il programma di un semplice convertitore A/D.

BIT ... E ALTRE COSE

Ci siamo spesso trovati a dover affrontare problemi connessi con i sistemi a microcalcolatore in cui esistono convertitori a diverso numero di bit. Se, infatti, in un sistema ci sono diverse risoluzioni sia per gli ingressi che per le uscite analogiche si possono usare insieme convertitori digitali-analogici e analogici-digitali a 8, 10 o 12 bit. Per esempio, i dati possono essere acquisiti con un convertitore A/D a otto bit e poi visualizzati con un convertitore D/A a 10 bit. In questo caso però sorge una domanda, che cosa facciamo con i bit? Nel caso in cui i diversi convertitori hanno lo stesso numero di bit, non esistono problemi, mentre nel caso in cui esiste un diverso numero di bit è necessario tenere in considerazione alcuni dati.

<u>INGRESSO MAGGIORE DELL'USCITA</u> Nel caso in cui il numero di bit provenienti dal convertitore A/D è maggiore del numero di bit del dato disponibili nel convertitore D/A che verrà usato per mettere in uscita l'informazione, non esistono grosse difficoltà nel trattare i dati.

Un tipico esempio è rappresentato da un sistema avente un convertitore A/D a 10 bit e un convertitore D/A a 8 bit, che viene usato per visualizzare o disegnare i dati. In questo caso è facile, per il calcolatore, "ignorare" i bit meno significativi della parola del dato a 10 bit. L'uscita del convertitore D/A non avrà una risoluzione grande quanto il convertitore A/D usato per acquisire dati. Anche se fossero ignorati i due bit meno significativi della parola a 10 bit, il segnale di uscita avrà ancora un errore minore del valore associato al bit meno significativo della parola a otto bit:

valore di ingresso del convertitore A/D a 10 bit 0111110011
valore di uscita del convertitore D/A a 8 bit 01111100

I due bit meno significativi della parola a 10 bit, 11, sono stati ignorati dal convertitore D/A. Questa manipolazione di bit è stata eseguita dal programma illustrato nell'Esempio 6-2 e 6-3.

ESEMPIO 6-2
ROUTINE PER CONVERTIRE DATI A 10 BIT IN DATI A 8 BIT PER USCITA D/A

```
                *003 100
003 100 170     MOVAB      PRENDE I DUE BIT PIU' SIGNIFICATIVI
003 101 037     RAR        RUOTA A DESTRA NEL CARRY
003 102 107     MOVBA      AGGIORNA IL RISULTATO
003 103 171     MOVAC      PRENDE GLI OTTO BIT MENO SIGNIFICATIVI
003 104 037     RAR        SPOSTA IL CARRY PRECEDENTE IN A
003 105 117     MOVCA      AGGIORNA IL RISULTATO
003 106 170     MOVAB      RIPETE PER IL BIT SUCCESSIVO
003 107 037     RAR
003 110 107     MOVBA
003 111 171     MOVAC
003 112 037     RAR
003 113 117     MOVCA
```

ESEMPIO 6-3
ROUTINE DI CONVERSIONE DI DATI A 10 BIT IN DATI A 8 BIT PER USCITA D/A, USANDO LE ISTRUZIONI DI MASCHERATURA E ROTAZIONE

```
                *003 100
003 100 171     MOVAC      PRENDE GLI OTTO BIT MENO SIGNIFICATIVI
003 101 017     RRC        RUOTA I DATI A DESTRA
003 102 017     RRC        LI RUOTA DI NUOVO
003 103 346     ANI        MASCHERA I DUE BIT MENO SIGNIFICATIVI CHE
003 104 077     077        SONO VECCHI, D0 e D1
```

```
003 105 117    MOVCA    LI MEMORIZZA RIPORTANDOLI NEL REGISTRO C
003 106 170    MOVAB    PRENDE I DUE BIT PIU' SIGNIFICATIVI
003 107 017    RRC      RUOTA ANCHE QUESTI
003 110 017    RRC
003 111 346    ANI      MASCHERA TUTTO TRANNE
003 112 300    300      I DUE BIT PIU' SIGNIFICATIVI
003 113 261    ORAC     ESEGUE FUNZIONE OR CON REGISTRO C
003 114 117    MOVCA    MEMORIZZA IL RISULTATO NEL REGISTRO C
```

Nell'Esempio 6-2 i dieci bit del dato sono stati semplicemente ruotati verso destra, eliminando i bit D1 e D0 della parola originaria, mentre il bit del carry viene utilizzato per trasferire i bit dal registro B dell'8080 al registro C durante le istruzioni di rotazione.

L'Esempio 6-3 funziona in modo più o meno simile, con la sola eccezione che le due parti a otto bit della parola a 10 bit vengono ruotate indipendentemente l'una dall'altra e infine combinate in una parola a otto bit ricorrendo a operazioni logiche. Va ricordato che nonostante siano stati usati solo due bit di una parola a otto bit, è l'intera parola a otto bit ad essere elaborata con le istruzioni logiche dell'8080. Questi due bit più significativi vengono memorizzati nel registro B dell'8080, bit D1 e D0, mentre gli otto bit che restano vengono memorizzati nel registro C.

Dovreste essere in grado di usare carta e penna per seguire la sequenza di passi di programma degli Esempi 6-2 e 6-3.

Se invece, vogliamo "arrotondare" il risultato a otto bit, in modo che rifletta con maggiore precisione la parola originaria a 10 bit che viene troncata,dobbiamo ricorrere ad una routine dal programma più complicato. Prima di esaminare il programma da usare, dobbiamo decidere esattamente ciò di cui abbiamo bisogno. Qui abbiamo scritto quattro numeri a 10 bit in cui 8 bit più significativi sono uguali:

0111110000   0111110001   0111110010   0111110011

Com'è possibile vedere, i due bit meno significativi possono presentarsi in qualunque delle quattro possibili condizioni, senza produrre degli effetti sugli otto bit più significativi. Per arrotondare questi ultimi, useremo le norme che seguono, basate sul valore dei due bit meno significativi.

Applicheremo cioé le regole seguenti:

Due BMS	Arrotondamento
0 0	Nessuno
0 1	Nessuno
1 0	Aggiunge 1 agli 8 BPS
1 1	Aggiunge 1 agli 8 BPS

Di fatto, il programma che esegue quanto detto è relativamente semplice, dato che l'arrotondamento sarà attuato semplicemente aggiungendo il valore 0000000010_2 al valore del dato a 10 bit. La somma esegue l'arrotondamento in tutti e quattro i casi:

```
  0111110000       01111110001       0111110010       0111110011
+ 0000000010     + 00000000010     + 0000000010     + 0000000010
  ----------       -----------       ----------       ----------
  0111110010       01111110011       0111110100       01111110101
```

Solo nei due casi in cui i due bit meno significativi sono 10 e 11 l'arrotondamento cambia effettivamente il valore degli otto bit più significativi. Non appena l'addizione è stata eseguita, il risultato a 10 bit può essere trattato come negli Esempi 6-2 e 6-3.

Questo arrotondamento risulta in modo più chiaro dove i valori del dato vengono acquisiti da una fonte a 10 bit e messi in uscita verso una destinazione a otto bit.

Esiste, tuttavia, un caso che ha valore limitativo e che richiede un test qualora si voglia eseguire un programma di arrotondamento. Supponiamo di aver avuto i valori 1111111110 o 1111111111. Avrebbero potuto essere arrotondati? Se fosse possibile, il valore a otto bit risultante sarebbe stato in entrambi i casi uguale a 00000000, con un superamento di capacità come risultato dell'addizione di 0000000010 con il dato. Ciò non è accettabile, dato che l'arrotondamento modifica di fatto i due valori massimi trasformandoli in zero.

Questo inconveniente può essere evitato testando un eventuale superamento di capacità che sarebbe causato dalla somma di 0000000010 sia a 1111111110 o 1111111111. Se si verifica un superamento di capacità, sarà indicato dall'11° bit o, se stiamo ancora usando il registro B e il registro C, dal bit D2 nel registro B. Il programma illustrato nell'Esempio 6-4 permette di effettuare l'arrotondamento, le rotazioni e il controllo dell'arrotondamento. Se viene localizzato un superamento, il risultato viene decrementato di uno per ripristinarlo a 11111111.

USCITA MAGGIORE DELL'INGRESSO Questo caso si verifica qualora il dispositivo di ingresso abbia una risoluzione inferiore a quella del dispositivo di uscita. Per esempio se il segnale che viene digitalizzato ha una risoluzione di otto bit, ed è necessario metterlo in uscita verso un convertitore D/A a 10 bit. I dati provenienti da un convertitore A/D a 10 bit vengono normalmente messi in uscita su un convertitore D/A a 10 bit sarebbe utile che nel nostro caso il convertitore A/D ad otto bit avesse lo stesso fondo scala di quello D/A a 10 bit. Questo semplificherebbe il confronto tra i vari valori.

ESEMPIO 6-4
ROUTINE DI ARROTONDAMENTO DI UN NUMERO A 10 BIT
DA METTERE IN USCITA VERSO UN CONVERTITORE D/A A 8 BT

```
                *003 100
003 100 003     INXB      AGGIUNGE 2 AL NUMERO A 10 BIT
003 101 003     INXB
003 102 170     MOVAB     QUI È UGUALE ALL'ESEMPIO 6-2
003 103 037     RAR       RUOTA IL NUMERO
003 104 107     MOVBA
003 105 171     MOVAC
003 106 037     RAR
003 107 117     MOVCA
003 110 170     MOVAB
003 111 037     RAR
003 112 107     MOVBA
003 113 171     MOVAC
003 114 037     RAR
003 115 117     MOVCA
003 116 170     MOVAB     PRENDE CIO' CHE È STATO LASCIATO DEI 2 BIT
                          PIU' SIGNIFICATIVI
003 117 037     RAR       RUOTA UN EVENTUALE RIPORTO
003 120 322     JNC       NEL BIT DEL CARRY
003 121 124     OK        SE NON C'È CARRY, L'ARROTONDAMENTO È
                          ESATTO
003 122 003     0
```

003 123 015		DCRC	DECREMENTA C SE L'ARROTONDAMENTO È ECCESSIVO
003 124 000	OK,	NOP	LA ROUTINE TERMINA QUI

Gli otto bit del dato generati dal convertitore A/D a otto bit possono essere posizionati in una qualsiasi di tre posizioni nell'ambito della parola a 10 bit. Daremo per scontato che i bit non verranno spezzati in due parti. I seguenti gruppi di parole a 10 bit illustrano i tre casi possibili:

00XXXXXXXX o 0XXXXXXXX0 o XXXXXXXX00

Qui le X rappresentano gli otto bit della parola a otto bit. Se il dato è messo in uscita verso il convertitore D/A a 10 bit come indicato nell'esempio a sinistra, l'uscita del convertitore D/A raggiungerà solo un quarto dell'uscita a scala intera, persino nel caso della più grande parola a otto bit, 11111111 dato che sarà messa in uscita come 0011111111.

Questo limite potrebbe essere facilmente superato installando un amplificatore con un guadagno di quattro sull'uscita del convertitore D/A, ma questo amplificherebbe anche l'uscita del valore a bit normali, e non è ammissibile.

Anche l'esempio nel mezzo presenta dei limiti, dato che il valore massimo verrebbe messo in uscita solo come metà dell'uscita a scala intera disponibile proveniente dal convertitore D/A a 10 bit.

Il migliore risultato si ottiene spostando a sinistra le due posizioni a otto bit avvicinandoci così al valore di fondo scala del convertitore D/A a 10 bit con una approssimazione pari a tre incrementi elementari.

Non dovrebbe essere un problema per voi spostare gli otto bit a sinistra per formare una parola a 10 bit. Le istruzioni di rotazione come quelle usate nell'Esempio 6-2 possono essere usate con la direzione di rotazione modificata da destra a sinistra.

L'uscita del dato a otto bit mantiene una precisione di una parte su 256 ed è ora direttamente comparabile ai segnali che hanno uguali tensioni di fondo scala, ma che sono state digitalizzate con una risoluzione a 10 bit.

CAPITOLO 7

ESPERIMENTI CON CONVERTITORI DIGITALI-ANALOGICI E ANALOGICO-DIGITALI

INTRODUZIONE AGLI ESPERIMENTI

Abbiamo deciso di dedicare un capitolo al convertitore A/D e agli esperimenti relativi. La ragione prinicipale è che vogliamo risparmiarvi un certo numero di esperimenti. Molti dei circuiti e dei passi di programma vengono usati in configurazioni leggermente differenti in una serie di esperimenti. Con gli esperimenti raggruppati in un solo capitolo, è facile passare da un esperimento all'altro senza dover smontare l'apparecchiatura solo per poi rimontarla nel capitolo successivo.

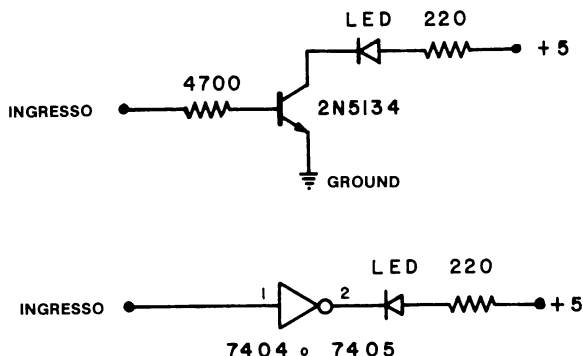
Fate attenzione alle note, poste in fondo al commento, di alcuni degli esperimenti che vi diranno che cosa va salvato e non rimosso dal piano di montaggio quando passate all'esperimento successivo. Se devono essere conservate alcune parti di programma, ricordate di lasciare l'apparecchiatura accesa.

A differenza di alcuni esperimenti realizzati nel Bugbook precedente, quelli contenuti in questo volume richiederanno un circuito di espansione su scheda per montaggio più ampio e alcuni programmi lunghi per poter funzionare in modo corretto. Va ricordato che è molto importante fare attenzione quando si sta caricando i programmi e si collegano i circuiti di interfaccia. Ciò permetterà di ridurre i tempi di debug sia del programma che del circuito, in caso di malfunzionamenti. Per semplificare parte della circuiteria, abbiamo sviluppato l'Outboard LR-35R, costituito da un convertitore digitale/analogico a 10 bit a doppia bufferizzazione con tutta la necessaria circuiteria di supporto. Il contenuto di questo capitolo sarà particolarmente utile per chi possiede zoccoli della piastra di montaggio molto diffusi come gli SK-10. Sia gli zoccoli della piastra di montaggio che gli Outboard LR-35 vengono forniti dalla E & L, Inc. Derby CT 06418.

Se preferite, invece, costruire o montare da soli un convertitore D/A a 10 bit, troverete il circuito completo dell'LR-35 nella Figura 7-1. In questo caso abbiamo utilizzato il convertitore D/A a 10 bit, Modello AD7522 dell'Analog Devices sull'LR-35, cui può essere sostituito, se lo si preferisce, un convertitore D/A a 10 bit e doppia bufferizzazione equivalente.

Molti degli esperimenti faranno riferimento specifico ai circuiti presentati negli esempi di interfacciamento. Comunque, volendo, si possono realizzare circuiti equivalenti nel caso in cui non si voglia rinunciare ad ottenere le parti utilizzate negli esempi. Gli esempi stessi e le figure riportate nel capitolo precedente forniranno gli elementi necessari per progettare qualcosa di specifico per i vostri impieghi.

In alcuni degli esperimenti qui riportati, vengono usate due porte di uscita, ciascuna delle quali è collegata a otto diodi LED, cosicché il dato binario può essere emesso a controllo di programma. Se questo tipo di porta in uscita non è disponibile nel vostro calcolatore, vi suggeriamo di realizzare alcuni circuiti "latch" utilizzando uno dei molti schemi relativi alle linee di uscita del *Bugbook III e Bugbook VI*. Alle due porte che sono state utilizzate è stato assegnato rispettivamente l'indirizzo 002 e 000 (02 e 00 esadecimale).

Il dispositivo di segnalazione, specificato nel disegno relativo all'Esperimento n. 3, può essere costituito da una parte di un LR-6 Lamp Monitor Outboard, o può essere un semplice LED con relativo circuito di pilotaggio. Due tipici esempi di impiego dei circuiti di segnalazione luminosa sono riportati nel seguito. Ambedue possono essere impiegati. Se si sceglie un circuito SN7404 o SN7405, il piedino 14 deve essere collegato a + 5 Volt e il piedino 7 a massa per fornire l'alimentazione al circuito integrato.

Gli esperimenti sono stati realizzati in modo da essere facilmente implementati sulla maggior parte dei sistemi a microcalcolatore basati sull'8080. Noi abbiamo utilizzato un sistema MMD-1 dell'E & L Instruments, che ha un bus dati bidirezionale a otto bit (non invertibile) e un bus di indirizzi unidirezionale a 16 bit. Alcuni calcolatori, come ad esempio l'SBC-80/10 dell'Intel, hanno un bus dati invertente. Alcuni sistemi ancora possono richiedere una circuiteria aggiuntiva per l'interfacciamento che non è riportata nei nostri esperimenti.

Nel corso di questo volume, abbiamo utilizzato la notazione $\overline{\text{IN}}$ per indicare un segnale di lettura da I/Q da parte del calcolatore e la notazione $\overline{\text{OUT}}$ per il segnale di scrittura. Questi segnali sono generati solo da comandi di trasferimento di I/O tramite accumulatore e sono equivalenti ai segnali $\overline{\text{I/OR}}$ e $\overline{\text{I/OW}}$ presenti in altri sistemi a microcalcolatore. Gli esperimenti usano gli indirizzi dei dispositivi di I/O 003, 004, 005 e 006. Il calcolatore MMS-1 che noi abbiamo utilizzato possiede una parte di decodifica relativa all'I/O che fornisce un facile accesso a questi indirizzi. Se il vostro calcolatore non possiede una logica di decodifica per dispositivi periferici che permetta tutto ciò, potete realizzare una semplice logica di decodifica così come riportata nel seguito. Il circuito permette infatti di generare i codici da 000 a 007. Se invece questi codici sono già presenti nel vostro microcalcolatore potete facilmente cambiare gli indirizzi dei dispositivi utilizzati nei programmi che sono riportati in ciascun esperimento.

Negli Esperimenti 7, 8 e 9 è stato utilizzato il modulo MP-10 della Burr-Brown. Questo dispositivo fornisce due convertitori D/A a otto bit in un solo contenitore. La logica di decodifica è fornita anch'essa all'interno del dispositivo; negli stessi esperimenti abbiamo utilizzato i codici 360, 361, e 363 per selezionare i dispositivi periferici. Nel caso in cui decidiate di utilizzare due convertitori a 8 bit al posto del modulo MP-10, dovete avere a disposizione un decodificatore per realizzare gli indirizzi opportuni. Sostituendo il circuito riportato nel seguito per quanto riguarda i cinque ingressi A3-A7 mostrati nello schema precedente che realizzava i codici 000 a 007, otterrete gli indirizzi da 360 a 367. Questi indirizzi sono riportati tra parentesi nello schema del circuito precedente.

Se la memoria a lettura/scrittura, che è a vostra disposizione, non ha una parte allocata agli indirizzi che vanno da 002 000 e 003 377 (0200 e 0FFF) avete due possibilità. Potete o selezionare diversamente gli indirizzi della vostra area di memoria a lettura/scrittura in modo da spostarla così che comprenda questi indirizzi, oppure potete cambiare gli indirizzi nei programmi in modo da renderli disponibili sull'area di memoria a vostra disposizione. Controllate attentamente i vostri programmi nel caso vogliate quest'ultima soluzione.

Abbiamo supposto che abbiate già stabilito un valore di puntatore di stack all'interno della CPU 8080. Questo può essere realizzato da un programma supervisore. Se invece ciò non viene realizzato si deve effettuare un caricamento del valore di stack; vi

suggeriamo di impiegare il valore 003 377 in modo da fissare l'area di stack nella parte superiore dell'area di memoria a lettura/scrittura utilizzata nel corso degli esperimenti. Se dovete eseguire il caricamento del puntatore di stack il metodo da seguire sarà il seguente:

1. Caricate prima nel vostro computer il programma che segue:

003 000 061	LXISP	Carica il puntatore di stack
003 001 377	377	Parte bassa dell'indirizzo di stack
003 002 003	003	Parte alta dell'indirizzo di stack
003 003 166	HLT	Stop

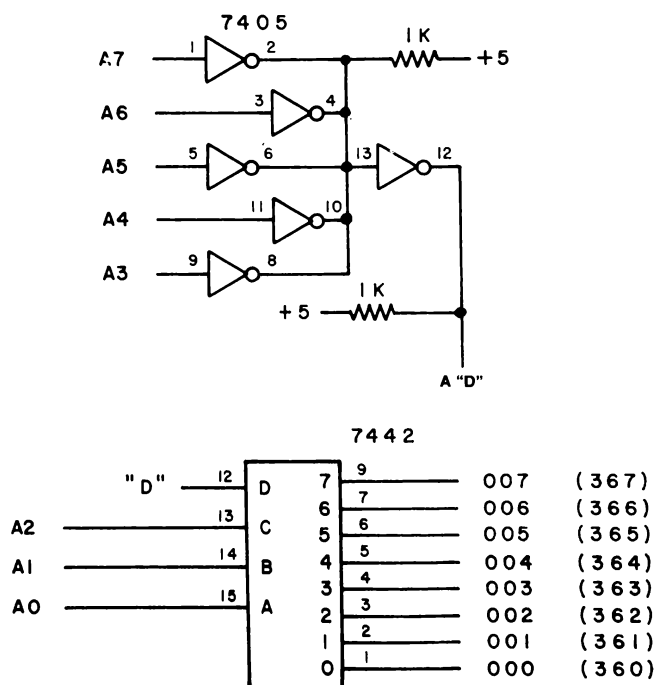
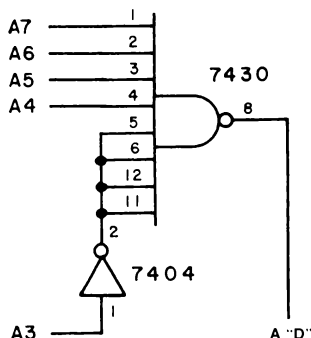
2. Eseguite il programma e quindi iniziate gli esperimenti. Ricordate di ripetere questa sequenza se spegnete e poi riaccendete il sistema.

Il computer MMD-1 che abbiamo usato per gli esperimenti fissa lo stack tramite l'impiego di un programma "Keyboard Executive".

Negli esperimenti che abbiamo eseguito sono stati utilizzati + 12 e − 12 volt, cioé due tensioni che non sono particolarmente pericolose, anche se la loro applicazione ad una parte sbagliata di una circuiteria può essere disastrosa. Per questa ragione, vi consigliamo di effettuare tutti i collegamenti all'alimentazione prima di tutti gli altri. Una volta eseguiti questi collegamenti, controllateli: potreste notare degli strani disturbi sulle uscite analogiche fornite dal convertitore D/A usato per i vostri esperimenti. Questi disturbi si noteranno in modo netto se userete anche uno oscilloscopio. Si tratta di disturbi dovuti al trasferimento di rumore digitale nel circuito analogico attraverso i collegamenti all'alimentazione da + 5, + 12, e − 12 volt. In questo caso si può ovviare all'inconveniente filtrando questi collegamenti con condensatori da 0,1 mfd posti tra il collegamento all'alimentazione e la terra.

7442

"D" — 12 — D
A2 — 13 — C
A1 — 14 — B
A0 — 15 — A

Uscita	Pin		
7	9	007	(367)
6	7	006	(366)
5	6	005	(365)
4	5	004	(364)
3	4	003	(363)
2	3	002	(362)
1	2	001	(361)
0	1	000	(360)

Se siete istruttori, sarete certamente interessati a inserire parti del programma in una memoria a sola lettura programmabile (PROM), che è particolarmente utile nelle classi numerose dato che riduce il tempo occorrente nelle operazioni di debug di circuiti e programmi di studenti.

Usate pure liberamente tutti i circuiti e i programmi e inventate tutti gli esperimenti che volete. Potreste anche arrivare a realizzare delle interessanti applicazioni dell'hardware e del software fornitovi.

Se non siete particolarmente esperti di piastre di montaggio, e avete bisogno di aiuto, consultate il *Bugbook V*, Capitolo 9. Scoprirete inoltre che un altro valido aiuto può essere il Breadboarding Station Outboard LR-25, che contiene tra l'altro otto visualizzatori, quattro interruttori, un clock R-C e due pulsanti.

Gli esempi che seguono hanno lo scopo di illustrare l'uso dei convertitori digitali/analogici e analogici/digitali.

Esperimento N.	Commento
1	Interfacciare un convertitore D/A a 10 bit con un calcolatore basato sull'8080 e usarlo per generare rampe di tensioni in uscita con periodi variabili.
2	Usare un convertitore D/A a 10 bit per generare forme d'uscita triangolare con periodi variabili. *
3	Costruire un convertitore A/D a rampa usando un convertitore D/A a 10 bit e un programma di controllo. *
4	Costruire un convertitore A/D ad approssimazioni successive usando un convertitore D/A a 10 bit e un programma di controllo. *
5	Costruire un piccolo sistema di acquisizione e visualizzazione di dati usando un convertitore A/D ad approssimazioni successive costruito nell'Esperimento 4. *
6	Misurare le tensioni di soglia di una porta NAND usando un convertitore D/A a 10 bit per mettere in uscita i valori di prova verso un dispositivo sottoposto a collaudo. *
7	Interfacciare il modulo MP-10 della Burr-Brown comprendente convertitori D/A con il computer e usarlo in modi diversi.
8	Usare convertitori A/D a otto bit per generare segnali necessari per la visualizzazione nel piano Y-T sull'oscilloscopio.
9	Usare convertitori D/A a otto bit per generare i segnali necessari ad una visualizzazione X-Y su un oscilloscopio.

*NOTA:** In questi esperimenti sono stati usati l'hardware e il software utilizzati nei precedenti esperimenti.

ESPERIMENTO N. 1

INTERFACCIAMENTO DI UN CONVERTITORE DIGITALE ANALOGICO A 10 BIT

Scopo

Lo scopo di questo esperimento è quello di interfacciare un convertitore digitale/analogico a 10 bit Modello AD7522 dell'Analog Devices con un microcalcolatore 8080. Per collaudare l'interfaccia verrà poi utilizzato un programma per uscita parallela.

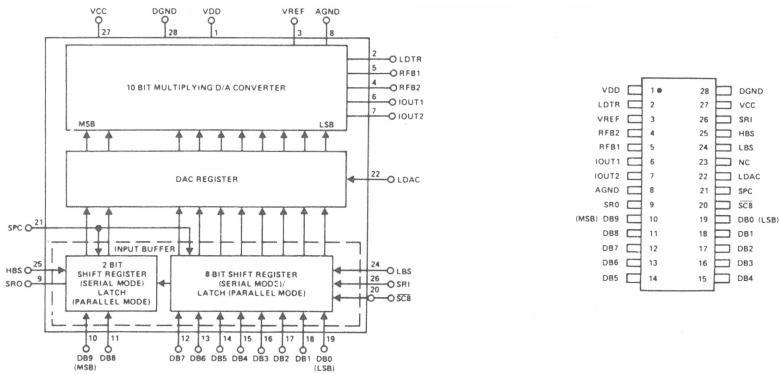
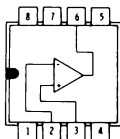
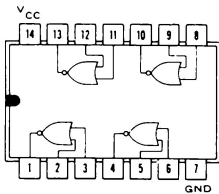
Commento

Il primo programma di questo esperimento utilizza un convertitore D/A a 10 bit Modello AD7522 e un convertitore D/A a 10 bit a doppia bufferizzazione equivalente, per generare una rampa lineare lenta che può essere osservata con l'aiuto di un multimetro (VOM) o di un oscilloscopio. L'andamento della informazione che viene messa in uscita verso il convertitore D/A verrà anch'essa visualizzata in corrispondenza delle due porte di uscita fornite di diodi LED.

Poiché il convertitore D/A AD7522 è un dispositivo a 10 bit, la tensione di uscita viene divisa in 1024 valori discreti, ciascuno dei quali è circa 5,5 millivolt più grande o più piccolo dei valori di tensione vicini. Il tempo di uno "sweep" completo di tutti i 1024 passi di tensione, viene determinato dal tempo necessario per eseguire un sottoprogramma di ritardo nell'ambito del software che genera la rampa lineare. Inizialmente il periodo di rampa sarà di circa 10 secondi.

Il convertitore D/A ha 10 ingressi di dati, uno per bit, mentre il bus di dati dell'8080 ha solo otto bit, per cui può fornire al convertitore D/A solo otto bit alla volta. Per la memorizzazione dei dati, l'AD7522 dispone di tre registri interni. Gli otto bit meno significativi di dati vengono caricati in un registro a otto bit quando viene generato l'impulso di strobe relativo alla parte bassa del byte (LBS) mentre i due bit più significativi vengono caricati in un registro a due bit, quando viene generato l'impulso di strobe relativo alla parte alta del byte (HBS). Il caricamento dei dati in questi due registri non produrrà effetti sulla tensione che viene messa in uscita dal convertitore D/A del dispositivo. Solamente nel caso di un terzo ingresso di caricamento del convertitore D/A (LDAC), l'intera parola a 10 bit verrà trasferita dai due registri provvisori al registro del convertitore D/A e applicata agli ingressi del convertitore D/A a 10 bit. Questa procedura è già stata esaminata in precedenza e viene chiamata *doppia bufferizzazione*. Nel caso in cui, invece, non disponiate di un convertitore D/A AD7522, dovrete costruire un'interfaccia di convertitore D/A a 10 bit e doppia bufferizzazione.

Nella prima parte del Capitolo 1 è illustrato lo schema dettagliato di un convertitore D/A a doppia bufferizzazione.

Configurazione dei pin dei circuiti integrati

CONVERTITORE D/A AD7522

741

7402

Schema del circuito

Nella prima parte avevamo illustrato lo schema di un convertitore D/A a 10 bit AD7522 completamente interfacciato, che riprendiamo nella Figura 7-1.

Se disponete di una scheda LR-35, potrete utilizzare quest'ultima invece di costruire il circuito di interfaccia per il circuito integrato AD7522, oppure potrete sostituirla con un qualsiasi convertitore D/A a 10 bit e doppia bufferizzazione.

Passo 1

Montare il circuito illustrato nella Figura 7-1 o usare la scheda LR-35 come indicato nella Figura 7-2. Se avete deciso di montare il circuito, come quello nella Figura 7-1, assicuratevi che siano eseguiti i collegamenti aggiuntivi indicati nella Figura 7-2.

Per osservare la tensione messa in uscita dal convertitore D/A, collegate un multimetro (VOM) al collegamento DAC OUT sulla piastra di montaggio. Poiché useremo una tecnica di I/O tramite accumulatore, assicuratevi che il segnale OUT dello 8080 sia collegato in modo corretto al punto etichettato con l'interfaccia o sulla scheda LR-35.

Useremo inoltre per gli indirizzi 003, 004 e 005. Se sul vostro calcolatore non sono direttamente disponibili gli indirizzi già decodificati, dovrete collegare il decodificatore di indirizzi più adatto. A questo proposito, troverete un circuito utile nell'introduzione di questo capitolo. Se usate un calcolatore MMD-1, scoprirete che i codici da 003 a 007

Figura 7-1. Schema di interfaccia di un convertitore D/A a 10 bit AD7522. Per un campo di uscita da 0 a + 5 Volt usare un regolatore di tensione 79L05 (VR-2).

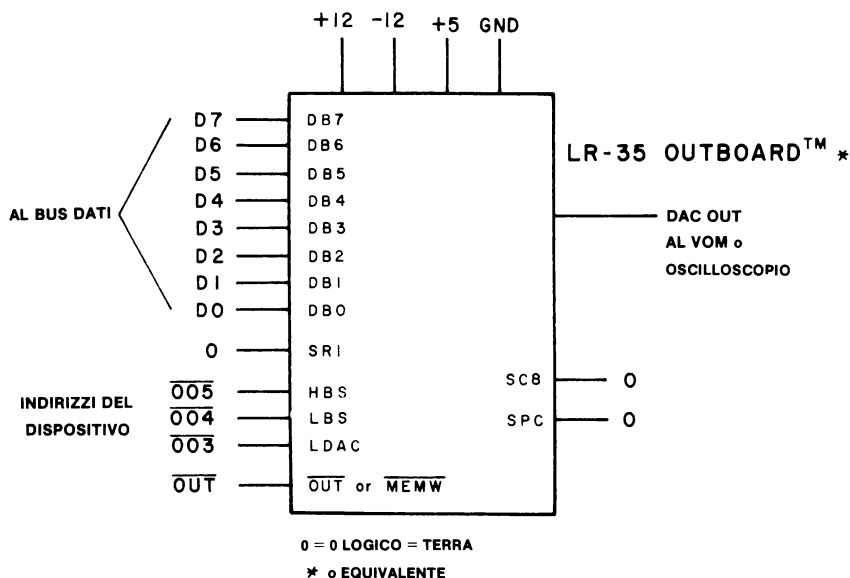



Figura 7-2. Collegamenti fra l'interfaccia AD7522 e il sistema a microcalcolatore 8080. Utilizzate questi collegamenti se avete una scheda LR-35.

sono sistemati sui piedini della piastra di montaggio vicini al circuito integrato del decodificatore SN74L42 situato nella parte DECODER I/O del circuito stampato principale.

Se avete dei dubbi sulla generazione degli impulsi di selezione del dispositivo e gli indirizzi del dispositivo di I/O consultate il Capitolo 17 del *Bugbook VI.*

Passo 2

Collegate l'alimentazione al vostro sistema di elaborazione e caricate il programma che segue nella memoria a lettura/scrittura del calcolatore. In questo esperimento useremo inoltre un sottoprogramma di ritardo che comincia dall'indirizzo 003 200 ed è illustrato anche nel listing che segue.

```
                             QUESTO PROGRAMMA UTILIZZA UN CONVERTITORE
                             ANALOGICO/DIGITALE A 10 BIT PER GENERARE UNA RAMPA
                             LINEARE

                             *003 000
003 000 175      START,      MOVAL        ACQUISISCE GLI 8 BIT MENO SIGNIFICATIVI DEI
003 001 323                  OUT          DATI E LI METTE IN USCITA VERSO IL REGISTRO
                                          PROVVISORIO
003 002 004                  004          CHE CONTIENE LA PARTE BASSA DEL BYTE DI
                                          DATI
003 003 323                  OUT          INOLTRE LI METTE IN USCITA VERSO UNA
                                          DELLE
003 004 002                  002          PORTE IN USCITA DEL CALCOLATORE
```

```
003 005 174           MOVAH     QUINDI ACQUISISCE GLI 8 BIT PIU'
                                SIGNIFICATIVI
003 006 323           OUT       E LI METTE IN USCITA VERSO IL DAC.
003 007 005           005
003 010 323           OUT       LI METTE INOLTRE IN USCITA VERSO UNA
                                PORTA DI
003 011 000           000       USCITA DOTATA DI DIODI LED
003 012 323           OUT       USA UN IMPULSO DI USCITA PER TRASFERIRE
003 013 003           003       I DATI DAL REGISTRO HOLDING DEL DAC.
                                QUINDI
003 014 315           CALL      CHIAMA UN SOTTOPROGRAMMA DI RITARDO.
003 015 200           DELAY
003 016 003           0
003 017 043           INXH      INCREMENTA IL VALORE DEL DATO DI 1
003 020 303           JMP       E METTE IN USCITA IL NUOVO VALORE.
003 021 000           START
003 022 003           0
                      SOTTOPROGRAMMA DI RITARDO

                      *003 200
003 200 365   DELAY,  PUSHPSW   SALVA I REGISTRI
003 201 325           PUSHD
003 202 021           LXID      PONE I BYTE DI TEMPORIZZAZIONE NEI
                                REGISTRI
003 203 046           046       D E E
003 204 001           001
003 205 033   DEC,    DCXD      DECREMENTA LA COPPIA DI REGISTRI
003 206 172           MOVAD
003 207 263           ORAE      CONTROLLA SE LA COPPIA DI REGISTRI = 000
003 210 302           JNZ       SE NON È ZERO, RIPETE L'OPERAZIONE
003 211 205           DEC
003 212 003           0
003 213 321           POPD      FATTO, RIPRISTINA I REGISTRI.
003 214 361           POPPSW
003 215 311           RET
```

Passo 3

Cominciate l'esecuzione del programma dall'indirizzo 003 000. Assicuratevi che il multimetro (VOM) sia collegato alla tensione messa in uscita dall'interfaccia del convertitore D/A, DAC OUT. Cosa osservate?

Noi abbiamo riscontrato un aumento lento ma costante della tensione messa in uscita dal convertitore D/A fino ad una lettura finale di circa + 5 Volt. A questo punto l'ago del VOM è ritornato a zero Volt e ha ripreso l'oscillazione. Il procedimento è stato ripetuto più volte e il tempo necessario per far oscillare la tensione da zero a + 5 Volt è stato, sul nostro calcolatore, di circa 10 secondi (clock di 750 kHz e periodi di 1,3 μsec).

Passo 4

Cambiate il contenuto della locazione della memoria 003 203 da 046 a 047 e date l'avvio al programma. Osservate qualche differenza tra il periodo necessario ora per un'oscillazione completa della tensione da zero a + 5 Volt e il periodo misurato prima?

Noi non abbiamo riscontrato alcuna differenza. Sapete perché?

È stato cambiato il byte del sottoprogramma di ritardo, ma solo per quanto riguarda il bit meno significativo, per cui il cambiamento è stato molto piccolo.

Passo 5

Cambiate il contenuto della locazione di memoria 003 204 da 001 a 002. Fate partire di nuovo il programma e annotate il tempo che occorre perché l'ago del VOM esegua un'escursione completa da zero a + 5 Volt.

Noi abbiamo notato che occorrono circa 18 secondi per far passare l'uscita del convertitore D/A da zero a + 5 Volt. Sapete perché il tempo di un'oscillazione completa da zero a + 5 Volt è aumentato in modo così considerevole rispetto a quello osservato nei Passi 3 e 4?

La grossa differenza del periodo della *rampa* della tensione è dovuta al cambiamento rilevante introdotto nei byte di temporizzazione usati dal sottoprogramma di ritardo. Tale cambiamento è stato effettuato sul byte più significativo dell'istruzione LXID a tre byte. Il primo cambiamento nel Passo 4 è stato quello di cambiare il byte di temporizzazione da 001 046 a 001 047.

Il secondo cambiamento, nel Passo 5, è consistito nell'aumentare il byte di temporizzazione da 001 047 in 002 047. *Ricordate che non abbiamo cambiato il byte meno significativo portandolo di nuovo a 046 dopo averlo mutato in 047 nel Passo 4.* Questo secondo cambiamento ha quasi raddoppiato il tempo impiegato dal sottoprogramma di ritardo.

Passo 6

Per generare una rampa positiva abbiamo utilizzato un convertitore D/A a 10 bit e un programma opportuno. Quale avrebbe potuto essere il metodo più semplice per generare una rampa negativa con l'attuale circuito di interfaccia?

Probabilmente la modifica più facile dovrebbe essere apportata al software. Noi vi consigliamo di sostituire una istruzione di decremento per l'istruzione di incremento usata ora nella locazione 003 017.

Cambiate l'istruzione nella locazione 003 017 da INXH (043) a DCXH (053) e cambiate i due byte di temporizzazione nel sottoprogramma di ritardo in modo che risultino uguali a quelli indicati nel listing del programma di DELAY, cioè, la locazione 003 203 dovrebbe contenere 046 e la locazione 003 204, 001.

Iniziate il programma dall'indirizzo 003 000 e annotate le vostre osservazioni sui cambiamenti della tensione del VOM nello spazio in alto della pagina a fianco.

Dovreste notare che l'ago inizia ora una lenta oscillazione verso il basso, da + 5 a zero Volt. Una volta che l'ago ha raggiunto lo zero, ritorna velocemente nella posizione corrispondente a + 5 Volt. Per poi ritornare di nuovo a zero. Il tempo di questo periodo di rampa dovrebbe aggirarsi sui 10 secondi dato che il sottoprogramma di ritardo è stato reinizializzato per contenere i byte di temporizzazione originali.

Passo 7

Cambiando il contenuto delle locazioni 003 014, 003 015 e 003 016 in 000, si elimina la chiamata al sottoprogramma di ritardo dal programma della rampa lineare. Che cosa provoca questo nel programma?

Tutti e tre i byte dell'istruzione di chiamata vengono sostituiti con istruzioni di "non operazione" (NOP = 000). Perché dobbiamo riempire tutti e tre i byte con questo tipo di istruzione? L'istruzione di chiamata potrebbe essere "eliminata" dal flusso di programma normale sostituendo semplicemente il primo byte con l'istruzione NOP?

Tutti e tre i byte devono essere "eliminati" dal programma. Se soltanto il primo byte di un'istruzione a più byte venisse sostituito con un'istruzione NOP, gli altri verrebbero trattati dal calcolatore come istruzioni. Nel programma della rampa lineare, sostituendo solo il primo byte dell'istruzione di chiamata con un'istruzione NOP si avrebbe il seguente risultato:

.	.	.	.
.	.	.	.
003	012	323	OUT
003	013	003	003
003	014	000	NOP
003	015	200	?
003	016	003	?
.	.	.	.
.	.	.	.

Il calcolatore non è in grado di sapere che si suppone che esegua le istruzioni rappresentate dai codici 200 e 003, la parte indirizzo dell'istruzione di chiamata. Quali operazioni rappresentano questi codici?

Il codice 200 è un'istruzione ADDB mentre il codice 003 è una istruzione INXB. *Quando dovete togliere un'istruzione da un programma, ricordate di sostituire tutti i byte con istruzioni NOP.*

Sostituite il contenuto delle locazioni 003 014, 003 015 e 003 016 con 000. Iniziate il programma dall'indirizzo 003 000. Cosa notate?

Nel nostro esperimento l'ago del VOM ha oscillato di circa 2,5 Volt. In termini di cambiamento di tensione le oscillazioni sono state piuttosto limitate. Poiché la rampa era ora generata senza il sottoprogramma di ritardo, questo periodo è risultato essere molto breve. Il VOM indica solo la tensione media corrente, per cui per osservare le rampe, sarà consigliabile l'uso di un oscilloscopio. Le rampe sono positive o negative?

Noi abbiamo osservato rampe negative. Ricordate che l'istruzione DCXH nel programma è stata sostituita.

NON DISCONNETTERE L'INTERFACCIA O SPEGNERE LA MACCHINA. L'hardware e parte del programma di questo esperimento saranno utilizzati nell'esperimento che segue.

ESPERIMENTO N.2

FORME D'ONDA COMPLESSE GENERATE DA CONVERTITORI DIGITALI-ANALOGICI

Scopo

Lo scopo di questo esperimento è di generare una forma d'onda triangolare impiegando il convertitore D/A che è stato interfacciato al microcalcolatore nel precedente esempio

Commento

Nel precedente esperimento abbiamo interfacciato il convertitore D/A a 10 bit AD7522 della Analog Devices con un microcalcolatore basato sull'8080. Inoltre un programma ha permesso di generare delle rampe di tensione sia positive che negative. Il cambiamento tra rampa positiva e negativa è stato semplice, in quanto è stato sufficiente sostituire un'istruzione di incremento (INXH) con una di decremento (DCXH). La generazione di una forma d'onda triangolare si ottiene facendo seguire una rampa positiva da una negativa, poi un'altra positiva e così via.

In questo esperimento incrementeremo un solo registro a otto bit. L'informazione incrementata viene emessa sia sugli otto bit meno significativi del convertitore D/A, sia su una porta di uscita in modo che il suo valore possa essere eventualmente letto tramite delle lampadine o dei visualizzatori a sette segmenti. Quando il contenuto del registro che è stato incrementato raggiunge il valore 377, il valore massimo possibile, il programma commuta su una serie di istruzioni che esegue il decremento del registro ed emette questi nuovi valori sul convertitore D/A e alla porta di uscita. Quando il contenuto del registro è decrementato fino a raggiungere il valore 000, i passi di programma che realizzano l'incremento permettono la generazione della parte positiva del segnale. Infine sarà di nuovo utilizzata, nel corso del programma, il sottoprogramma di ritardo DELAY.

Configurazione del pin del circuito integrato

Schema del circuito

Sia la configurazione dei circuiti integrati che lo schema generale del circuito sono stati riportati nell'Esperimento N. 1. Per ulteriori dettagli, esaminare quelli riportati precedentemente.

Passo 1

Se il circuito di interfaccia sviluppato nell'Esperimento N. 1 non è presente nel vostro calcolatore, si consiglia di vedere i dettagli riportati nel primo passo dell'Esperimento N. 1 e di ricollegare il convertitore come indicato. Se il circuito è intatto, procedere oltre con il Passo 2.

Passo 2

Immettere il programma che segue nella memoria a lettura/scrittura e partire dall'indirizzo 003 000. Questo programma utilizza il sottoprogramma di ritardo che è stato caricato in memoria durante l'Esperimento N. 1. Nel caso in cui questo sottoprogramma non sia presente nella memoria del vostro calcolatore, è stato riportato nel listato che segue e va caricato a partire dall'indirizzo 003 200.

```
                         QUESTO PROGRAMMA GENERA UNA FORMA D'ONDA
                         TRIANGOLARE E L'USO DI UN CONVERTITORE DIGITALE-
                         ANALOGICO

                         *003 000
003 000 074   UP,        INRA        INCREMENTA IL CONTENUTO DEL REGISTRO A
003 001 312              JZ          SE IL VALORE È 000, È STATO OTTENUTO DOPO
                                     IL
003 002 020              DOWN        VALORE 377, COSI' SI DEVE A = 376
003 003 003              0           E SUCCESSIVAMENTE INIZIARE A
                                     DECREMENTARE
003 004 323              OUT         CARICA IL DATO NEL REGISTRO
003 005 004              004         PROVVISORIO DEL DAC
003 006 323              OUT         ED ANCHE A UNA DELLE PORTE IN USCITA
003 007 002              002         DEL CALCOLATORE COLLEGATE AI LED
003 010 323              OUT         QUESTO IMPULSO TRASFERISCE IL
                                     CONTENUTO DEL
003 011 003              003         REGISTRO PROVVISORIO AL DAC
003 012 315              CALL        CHIAMATA AL SOTTOPROGRAMMA DI
                                     RITARDO
003 013 200              DELAY       IN MODO DA VEDERE IL CAMBIAMENTO SUL
                                     VOM
003 014 003              0           E SUI LED
003 015 303              JMP
003 016 000              UP          ORA INCREMENTA A E
003 017 003              0           RIPETE LA PROCEDURA
003 020 076   DOWN,      MVIA        PER ULTIMO È STATO IN USCITA IL VALORE 377,
                                     QUINDI
003 021 376              376         METTE IN USCITA IL VALORE 376 PER LA RAMPA
                                     NEGATIVA
003 022 323              OUT         METTE IN USCITA IL DATO
003 023 004              004         SUL REGISTRO PROVVISORIO DAC
003 024 323              OUT         E SULLA PORTA DI USCITA
003 025 002              002         DEL CALCOLATORE FORNITA DI LED
003 026 323              OUT         FORNISCE L'IMPULSO PER MEMORIZZARE IL
                                     CONTENUTO DEL REGISTRO
003 027 003              003         PROVVISORIO NEL DAC
003 030 075              DCRA        DECREMENTA IL CONTENUTO DI A
003 031 315              CALL        ORA ATTENDE CHE IL VOM SI ASSESTI
003 032 200              DELAY       A CAUSA DEL LENTO TEMPO DI RISPOSTA
003 033 003              0
003 034 302              JNZ         ABBIAMO DECREMENTATO A AD 000?
003 035 022              DOWN + 2    NO? CONTINUIAMO A VISUALIZZARE E
                                     DECREMENTARE
003 036 003              0
003 037 303              JMP         SI? ALLORA RICOMINCIA LA RAMPA POSITIVA
003 040 000              UP
003 041 003              0

                         SOTTOPROGRAMMA DI RITARDO

                         *003 200
003 200 365   DELAY,     PUSHPSW     SALVA I REGISTRI
003 201 325              PUSHD
003 202 021              LXID        METTE I BYTE DI TEMPORIZZAZIONE NEI
                                     REGISTRI
003 203 046              046         D ED E
003 204 001              001
003 205 033   DEC,       DCXD        DECREMENTA LA COPPIA DI REGISTRI
003 206 172              MOVAD
```

003 207 263	ORAE	CONTROLLA CHE LA COPPIA DI REGISTRI SIA = 000
003 210 302	JNZ	SE NON È UGUALE A ZERO RIPETE L'OPERAZIONE
003 211 205	DEC	
003 212 003	0	
003 213 321	POPD	ESEGUITO. RIPRISTINA I REGISTRI.
003 214 361	POPPSW	
003 215 311	RET	

Passo 3

Fate partire il programma dall'indirizzo 003 000 e prendete nota dei valori di tensione più alto e più basso sul VOM mentre il programma è in esecuzione:

Noi abbiamo notato che l'ago del VOM oscilla tra 0 e 1,3 Volt, cosa che non necessariamente dovete notare anche voi.

Tuttavia, viene naturale una domanda; perché l'ago non esegue una escursione completa tra 0 e + 5 Volt, come nel caso dell'esperimento di generazione di rampa lineare?

Ricordate che l'AD7522 è un convertitore digitale-analogico a 10 bit e che noi, in questo esperimento, stiamo usando gli otto bit meno significativi. È impossibile conoscere lo stato dei due bit più significativi del vostro convertitore. Il software non li ha modificati, per cui, chi esegue questo esperimento può notare quattro possibili campi di tensione:

Campo	Due byte più significativi	Otto byte meno significativi	
		Minimo	Massimo
0,00-1,25 V	00	00000000	11111111
1,25-2,50 V	01	00000000	11111111
2,50-3,75 V	10	00000000	11111111
3,75-5,00 V	11	00000000	11111111

Dovreste notare che l'ago del vostro VOM oscilla con una differenza tra 1,2 e 1,4 circa, approssimativamente vicino ad uno dei campi sopra elencati.

Passo 4

Ora cambiate il periodo di ritardo del sottoprogramma di DELAY. Cambiate il valore contenuto nella locazione di memoria 003 204 da 001 a 000, fate partire il programma dall'indirizzo 003 000 e prendete nota di qualsiasi differenza di frequenza dell'uscita di tensione a triangolo, come risulta dal VOM e sui LED della porta di uscita.

Noi abbiamo notato che la frequenza era maggiore (periodo più breve), e il motivo è che il sottoprogramma di DEALY (ritardo) ha utilizzato valori di dati diversi. Cambiando i dati di temporizzazione da 001 046 a 000 046 dovreste osservare che in 15 secondi si verificano circa 23 cicli completi dell'uscita triangolare.

Passo 5

Un altro modo di rendere più veloce la frequenza dell'uscita di tensione triangolare consiste nel cambiare l'istruzione PUSHPSW del sottoprogramma di DELAY in un'istruzione RET (311). Con quali conseguenze?

Cambiando l'istruzione PUSHPSW (365) in un'istruzione RET (311), o istruzione di ritorno, si ottiene che il calcolatore ritorni immediatamente dal sottoprogramma una volta che la chiamata al sottoprogramma di DELAY è stata eseguita, per cui non si ha l'esecuzione dei restanti passi del sottoprogramma di DELAY.

Passo 6

Cambiate l'istruzione contenuta nella locazione di memoria 003 200 da 365 a 311. Ciò inserisce l'istruzione di ritorno all'inizio del sottoprogramma di DELAY. Eseguite il programma e annotate che cosa osservate.

Noi abbiamo dovuto usare un oscilloscopio per osservare la rampa dato che aveva una frequenza di circa 20 Hz, ed è stato difficile osservare dei cambiamenti di tensione così veloci sul VOM o in corrispondenza della porta di uscita.

Nella locazione di memoria 003 200 sostituite l'istruzione di ritorno con l'istruzione PUSHPSW (365) che c'era in precedenza, e sostituite, nella locazione di memoria 003 204, il byte di temporizzazione originale (001).

NON DISCONNETTERE L'INTERFACCIA O SPEGNERE LA MACCHINA.
L'hardware e parte del programma di questo esperimento saranno utilizzati nell'esperimento che segue.

ESPERIMENTO N.3

CONVERTITORE A/D A RAMPA A CONTROLLO DI PROGRAMMA

Scopo

Lo scopo di questo esperimento è di costruire un convertitore analogico/digitale usando il convertitore digitale/analogico a 10 bit, interfacciato in precedenza con il calcolatore assieme al comparatore LM311, e una porta di ingresso. In questo esperimento per determinare il valore di tensioni analogiche sconosciute, verrà impiegata la tecnica della conversione a rampa lineare.

Commento

L'esperimento si basa sull'impiego di una rampa lineare lenta, simile a quella generata durante l'Esperimento 1. L'ingresso di tensione sconosciuta, $V_{UNKNOWN}$, viene continuamente comparata con la rampa di tensione lineare che viene messa in uscita dal convertitore D/A a controllo di programma. Se l'uscita del convertitore D/A, V_{DAC}, o DACOUT, è inferiore all'ingresso di tensione sconosciuta, lo stato del comparatore sarà 1 o 0 logico; quando, invece, V_{DAC} sarà maggiore, l'uscita del comparatore sarà un 1 logico. Lo stato dell'uscita del comparatore viene caricato nel registro A dell'8080 tramite una porta di ingresso three-state. Viene poi testato mediante un'istruzione jump condizionale, JNZ. Se l'uscita del comparatore è lo 0 logico nella rampa, viene generato un altro passo (o gradino?) di tensione, se invece è uguale a un 1 logico, la tensione sconosciuta è stata raggiunta e il procedimento viene ripetuto iniziando la rampa a zero volt.

I registri H e L dell'8080 conservano i valori binari usati per generare la rampa. I dati vengono messi in uscita verso il convertitore D/A a 10 bit e verso le due porte di uscita, dove i valori binari possono essere visualizzati. L'uscita del comparatore può essere osservata con un dispositivo di visualizzazione di cui un tipico circuito è illustrato nell'introduzione a questi esperimenti.

Configurazione dei pin del circuito integrato

8095 o 74365

311

Schema del circuito

Passo 1

Collegare la circuiteria addizionale illustrata nel diagramma sopra. In questo esperimento useremo di nuovo il convertitore D/A a 10 bit AD7522 interfacciato con il computer dell'Esperimento 1. Se questo convertitore non è interfacciato al vostro computer, vi rimandiamo all'Esperimento 1, Passo 1, per i necessari dettagli sull'interfacciamento.

Ricordate di collegare l'uscita del convertitore D/A, V_{DAC}, o DACOUT, al collegamento V_{DAC} indicato nello schema contenuto in questa pagina. Per osservare l'uscita di tensione del convertitore D/A, si può utilizzare il VOM, anche se non è indispensabile.

Passo 2

Immettete il programma che segue nella memoria del calcolatore iniziando dall'inizio 003 000. Il sottoprogramma di DELAY che è stato usato negli esperimenti precedenti verrà usato anche in questo. Assicuratevi che sia caricato in modo corretto, iniziando dall'indirizzo 003 220. Il sottoprogramma di visualizzazione dei dati, DISPLA, si trova "sopra" il sottoprogramma di DELAY e inizia dall'indirizzo 003 220. Fare attenzione durante la procedura di caricamento di questi passi di programma.

```
                          QUESTO PROGRAMMA UTILIZZA UN DAC E UN
                          COMPARATORE PER REALIZZARE UN CONVERTITORE
                          ANALOGICO/DIGITALE A RAMPA

                          *003 000
003 000 041     ADC,      LXIH       I REGISTRI H ED L VENGONO USATI PER IL
                                     VALORE
003 001 000               000        CORRENTE IN RAMPA DIGITALE
003 002 000               000        INIZIALMENTE VENGONO POSTI ENTRAMBI A
                                     000
003 003 323               OUT        L'ISTRUZIONE OUT VIENE USATA PER
                                     GENERARE
003 004 007               007        UN IMPULSO DI SINCRONIZZAZIONE PER UN
                                     OSCILLOSCOPIO
```

```
003 005 175     DACOUT, MOVAL     PRENDE GLI OTTO BIT DI RAMPA MENO
                                  SIGNIFICATIVI
003 006 323             OUT       LI PONE IN USCITA SUL CONVERTITORE D/A
003 007 004             004
003 010 174             MOVAH     PRENDE GLI OTTO BIT DI RAMPA PIU'
                                  SIGNIFICATIVI
003 011 323             OUT       LI METTE IN USCITA SUL CONVERTITORE D/A
003 012 005             005
003 013 323             OUT       GENERA L'IMPULSO DI MEMORIZZAZIONE SUL
                                  REGISTRO PROVVISORIO DEL CONVERTITORE
003 014 003             003
003 015 315             CALL      CHIAMA IL SOTTOPROGRAMMA DI
                                  VISUALIZZAZIONE DEI REGISTRI H&L
003 016 220             DISPLA    E LA RELATIVA ROUTINE DI DELAY IN MODO DA
                                  POTER
003 017 003             0         VEDERE IL CONTENUTO DI H&L
003 020 333             IN        ORA CONTROLLA L'USCITA DEL
                                  COMPARATORE
003 021 006             006       PER VEDERE SE H&L HANNO RAGGIUNTO UN
                                  VALORE PARI
003 022 346             ANI       ALLA TENSIONE ANALOGICA SCONOSCIUTA
003 023 200             200
003 024 302             JNZ       H&L SONO GRANDI ABBASTANZA PERCHE'
003 025 000             ADC       L'USCITA DEL COMPARATORE È UGUALE A 1,
                                  QUINDI
003 026 003             0         INIZIARE UN'ALTRA CONVERSIONE
003 027 043             INXH      H&L NON SONO SUFFICIENTEMENTE GRANDI,
                                  PERCIO' INCREMENTARE
003 030 303             JMP       ENTRAMBI DI 1 E RITENTARE
003 031 005             DACOUT
003 032 003             0

                        ECCO UN PROGRAMMA DI VISUALIZZAZIONE IN USCITA

                        *003 220
003 220 174     DISPLA, MOVAH     PRENDE LA PARTE ALTA DEL BYTE DI DATI
003 221 323             OUT       E MEMORIZZA IL VALORE SU UNA DELLE
003 222 000             000       PORTE DI USCITA A LED SUL COMPUTER
003 223 175             MOVAL     PRENDE LA PARTE BASSA DEL BYTE DI DATI E
003 224 323             OUT       VISUALIZZA ANCHE QUESTA
003 225 002             002
003 226 315             CALL      DOPO AVER VISUALIZZATO H&L, RITARDA PER
003 227 200             DELAY     POCHI MSEC O SEC.
003 230 003             0
003 231 311             RET       QUINDI RITORNA AL PROGRAMMA PRINCIPALE

                        SOTTOPROGRAMMA DI RITARDO

                        *003 200
003 200 365     DELAY,  PUSHPSW   SALVA I REGISTRI
003 201 325             PUSHD
003 202 021             LXID      PONE I BYTE DI TEMPORIZZAZIONE NEI
                                  REGISTRI
003 203 046             046       D ED E
003 204 001             001
003 205 033     DEC,    DCXD      DECREMENTA LA COPPIA DI REGISTRI
003 206 172             MOVAD
003 207 263             ORAE      CONTROLLA CHE LA COPPIA DI REGISTRI SIA =
                                  000
003 210 302             JNZ       SE NON È ZERO RITENTA
003 211 205             DEC
```

003 212 003	0	
003 213 321	POPD	ESEGUITO, RIPRISTINA I REGISTRI
003 214 361	POPPSW	
003 215 311	RET	

Passo 3

Fate partire il programma dall'indirizzo 003 000. Che cosa osservate in corrispondenza dei LED collegati con le porte di uscita 000 e 002? Che cosa avviene in corrispondenza del VOM? Variate la posizione del potenziometro e annotate qui di seguito le vostre osservazioni:

Da parte nostra abbiamo osservato che ruotano il potenziometro verso un'estremità, un numero sempre minore di LED si illuminava in corrispondenza delle porte di uscita. Anche la tensione indicata dal VOM diventava sempre più bassa e il monitor (visualizzatore) collegato all'uscita del comparatore tremava debolmente.

Non appena il potenziometro veniva ruotato verso l'estremità opposta, si illuminava al contrario un numero sempre maggiore di LED in corrispondenza delle porte di uscita 000 e 002. L'ago del VOM si portava su valori di tensioni più alti e non appena questi venivano raggiunti, i LED in corrispondenza delle porte di uscita 000 e 002 aumentavano e tremavano. Sapete spiegare qual'è la ragione di questo tremolio?

Occorre del tempo perché il calcolatore effettui l'incremento e la visualizzazione del valore contenuto nei registri H&L, soprattutto in quanto viene interrotto un sottoprogramma di ritardo. Il tremolio è prodotto dall'intervallo di tempo occorrente al calcolatore per incrementare il contenuto di H&L di volta in volta.

Il computer impiegherà più tempo a misurare una tensione alta piuttosto che una bassa tensione?

Sì, poiché il conteggio necessario ad un convertitore D/A per generare una tensione alta richiede un programma più lungo di quello richiesto da un conteggio relativamente basso per basse tensioni.

Passo 4

Togliete il programma di ritardo dal flusso del programma normale, sostituendo un'istruzione di ritorno (311) all'istruzione contenuta nella locazione di memoria 003 200. Fate partire di nuovo il programma. Ruotate il potenziometro e notate l'effetto sulle tensioni misurate dal VOM. Iniziate con una tensione bassa (pochi LED illuminati) e procedete verso tensioni sempre più alte. Che cosa osservate?

Noi abbiamo notato che quando il potenziometro presenta al comparatore tensioni relativamente basse, l'ago del VOM rimane relativamente fermo. Non appena la tensione

viene aumentata di circa 2 volt, l'ago comincia ad oscillare o tremare. Quando mediante il potenziometro viene applicata al comparatore una tensione più elevata, l'ago del VOM comincia ad oscillare con un'ampiezza di mezzo volt. Pensate che sia naturale che ciò avvenga?

Si, ancora una volta le oscillazioni sono causate da una risposta lenta del VOM alle variazioni di tensioni. Il calcolatore può generare la rampa di tensione messa in uscita dal convertitore D/A più in fretta di quanto il VOM non riesca a misurarla. Ricordate che non esiste ritardo di tempo nel programma.
Prima di andare avanti, sostituite l'istruzione di ritorno che avete memorizzato nella locazione 003 200 con l'istruzione PUSHPSW (365). Ciò serve a ripristinare l'operazione del sottoprogramma di ritardo, DELAY.

Passo 5

Modificheremo ora il sottoprogramma di DELAY per rallentare il processo di conversione; ciò si ottiene semplicemente cambiando i byte di temporizzazione nelle locazioni 003 203 e 003 204. Dovrete dunque effettuare le seguenti modifiche al sottoprogramma di DELAY. Cambiate il valore contenuto nella locazione di memoria 003 203 trasformandolo in 000 e il valore della locazione 003 304 in 050. Fate partire il programma dell'indirizzo 003 000 e annotate le vostre osservazioni.

Il computer incrementa ora il valore di rampa alla velocità di tre conteggi al secondo. Come abbiamo già osservato in precedenza, i valori visualizzati sui LED diventano sempre più piccoli a mano a mano che il potenziometro presenta tensioni sempre più basse al comparatore, mentre, non appena la tensione viene aumentata girando il potenziometro nella direzione opposta, i valori visualizzati sui LED diventano sempre più alti.
Dovreste inoltre essere in grado di vedere l'uscita del visualizzatore cambiare stato, quando l'uscita del comparatore indica che la tensione di rampa è uguale o è maggiore dell'ingresso sconosciuto, $V_{UNKNOWN}$. Notate che l'uscita del comparatore ritornerà allo zero logico quando la rampa viene fatta ripartire dal suo valore più basso, 0000000000_2.

Passo 6

Se avete a disposizione un oscilloscopio a due tracce, potrete realizzare quanto descritto in questo passo. Se non disponete di questo strumento, proseguite al Passo 7.
Collegate uno degli ingressi dell'oscilloscopio all'uscita del convertitore D/A V_{DAC} o DACOUT, quindi collegate l'altro ingresso dell'oscilloscopio all'uscita $V_{UNKNOWN}$.

Togliete la chiamata al sottoprogramma di DELAY inserendo tre istruzioni NOP (000) nelle locazioni 003 226, 003 227 e 003 230. Ciò accellererà la generazione della rampa. Regolate il potenziometro nel suo punto medio, iniziate il programma e osservate la rampa sull'oscilloscopio. Usando la seconda traccia, posizionate la linea prodotta dal $V_{UNKNOWN}$ in cima alla rampa. Fate attenzione che entrambi i canali di ingresso siano posti sugli stessi livelli di guadagno e che siano in condizioni di accettare i livelli di ingresso d.c.

Che cosa osservate quando variate il valore imposto dal potenziometro?

Dovreste osservare due cose importanti. La prima è che la cima della rampa dovrebbe essere uguale alla tensione sconosciuta fino all'ampiezza massima dell'uscita del

convertitore D/A. La seconda cosa importante è che non appena la tensione sconosciuta si abbassa, si ha un numero di conversioni per unità, maggiore, di tempo. Le tensioni più alte richiedono tempi di conversione più lunghi.

Passo 7

Il programma che state usando per il convertitore A/D a rampa controllato da programma, visualizza costantemente i vari valori di test che vengono anche messi in uscita verso il convertitore D/A a 10 bit. Questo programma può essere modificato in modo che solo il risultato finale della conversione venga visualizzata in corrispondenza delle porte di uscita 000 e 002.
Siete in grado di suggerire le modifiche di programma necessarie a raggiungere questo obiettivo? Ricordate che i dati binari devono essere messi in uscita solo dopo che è stata completata la conversione.

Ecco le modifiche che suggeriamo:

a) Togliere la chiamata al sottoprogramma DISPLA, sostituire con le istruzioni NOP il contenuto delle locazioni 003 015, 003 016 e 003 017.

b) L'istruzione JNZ all'indirizzo 003 024 viene eseguita solo alla fine dalla conversione. Eseguire questa modifica:

da		a		
	JNZ		JNZ	302
	ADC		DISPLA	220
	∅		∅	003

Ciò fa sì che il programma visualizzi i dati alla fine della conversione.

c) Il programma DISPLA non viene più chiamato come sottoprogramma per cui l'istruzione return non è più necessaria. Sostituitela con un salto all'indirizzo simbolico ADC. Ciò farà sì che, dopo la visualizzazione dei dati, si abbia l'inizio di una nuova conversione:

003 231	JMP	303
003 232	ADC	000
003 233	∅	003

Effettuate queste modifiche ed eseguite il programma. A questo punto dovreste osservare solo il valore finale della tensione sconosciuta digitalizzata e non i valori di test, di rampa durante l'incremento sulle porte di uscita 000 e 002.

Ricordate di inserire l'istruzione di ritorno (311) nella locazione 003 231 prima di proseguire con l'esperimento che segue.

CONCLUSIONI

Abbiamo visto che si può costruire un convertitore analogico/digitale usando un convertitore digitale/analogico, un comparatore e di un microcalcolatore per una funzione decisionale e di controllo. Quando, tuttavia, si usa la tecnica della conversione a rampa, abbiamo osservato che il tempo di conversione è direttamente proporzionale alla tensione che si deve misurare o convertire.

L'esperimento che segue vi mostrerà come si può costruire un convertitore ad approssimazioni successive usando lo stesso circuito utilizzato in questo esperimento e cambiando solamente il programma.

NON DISCONNETTERE L'INTERFACCIA O SPEGNERE LA MACCHINA. Il circuito e parte del programma di questo esperimento saranno utilizzati nell'esperimento che segue.

ESPERIMENTO N.4

CONVERTITORE A/D AD APPROSSIMAZIONI SUCCESSIVE A CONTROLLO DI PROGRAMMA

Scopo

Lo scopo di questo esperimento è di usare la tecnica della conversione analogica/digitale ad approssimazioni successive per digitalizzare un segnale analogico.

Commento

La tecnica della conversione analogica/digitale ad approssimazioni successive sottopone a test ciascuna delle posizioni dei bit binari singoli in sequenza, dal più significativo (MSB) al meno significativo (LSB). Il processo inizia con l'applicazione di un uno logico al convertitore D/A nella posizione bit D9 (MSB bit più significativo) e mettendo a confronto la tensione di prova risultante con la tensione sconosciuta servendosi di un comparatore. Se la tensione da misurare è inferiore alla tensione di collaudo, il bit è riportato allo stato zero logico e viene sottoposto a test il bit successivo. Se invece la tensione sconosciuta non viene superata da questo test, il bit rimane allo stato di uno logico mentre si procede al test del bit meno significativo, usando la stessa procedura.

In questo modo il collaudo procede dal valore massimo a quello minimo relativamente ad ogni posizione. Le singole posizioni dei bit e le tensioni di prova corrispondenti per un convertitore D/A a 10 bit con una scala completa da zero a + 5,115 volt, sono elencate qui sotto:

Posizione del bit	Valore di tensione relativo
D0	0,005 Volts
D1	0,010
D2	0,020
D3	0,040
D4	0,080
D5	0,160
D6	0,320
D7	0,640
D8	1,280
D9	2,560

Schema del circuito

Il circuito usato nell'Esperimento 4 è uguale a quello usato nell'Esperimento 3. Per maggiori dettagli, pertanto, vi rimandiamo a quell'esperimento. Per quanto riguarda l'interfaccia con un convertitore D/A a 10 bit, impiegata in questo esercizio, consultate il Passo 1 nell'Esperimento 1.

Passo 1

Caricate il programma che segue nella memoria del calcolatore iniziando dall'indirizzo 003 000. Ricordate che questo programma utilizzerà anche i sottoprogrammi DELAY e DISPLA usati nell'esperimento precedente. Se questi ultimi saranno ancora disponibili nella memoria a lettura/scrittura del calcolatore potrete usarli, non senza averli prima controllati. Se invece dovete nuovamente introdurli, li troverete nel listing che segue.

```
                           QUESTO PROGRAMMA UTILIZZA LA TECNICA
                           DELL'APPROSSIMAZIONE SUCCESSIVA
                           PER DETERMINARE IL VALORE DIGITALE
                           DI UNA TENSIONE ANALOGICA SCONOSCIUTA, USANDO UN
                           CONVERTITORE D/A AD7522 E UN COMPARATORE

                   *003 000
003 000 041  ADC,    LXIH    H&L CONTENGONO L'APPROSSIMAZIONE
                             CORRENTE
003 001 000          000
003 002 002          002     CHE È XX XXX X10 00 000 000
003 003 021          LXID    D&E CONTENGONO IL BIT CORRENTE
003 004 000          000     CHE È SOMMATO O SOTTRATTO
003 005 002          002     DALL'APPROSSIMAZIONE CORRENTE
003 006 006          MVIB    B È USATO COME CONTATORE DI BIT
003 007 012          012     012 OTTALE = 10 DECIMALE (CONVERTITORE
                             D/A A 10 BIT)
003 010 175  DACOUT, MOVAL   PRENDE GLI OTTO BIT DI DATI MENO
                             SIGNIFICATIVI
003 011 323          OUT     LI METTE IN USCITA VERSO IL CONVERTITORE
                             D/A
003 012 004          004
003 013 174          MOVAH   PRENDE I DUE BIT DI DATI PIU' SIGNIFICATIVI
003 014 323          OUT     LI METTE IN USCITA VERSO IL CONVERTITORE
                             D/A
003 015 005          005
003 016 323          OUT     SUL REGISTRO PROVVISORIO DEL
                             CONVERTITORE
003 017 003          003
003 020 315          CALL    VISUALIZZA L'APPROSSIMAZIONE CORRENTE
003 021 220          DISPLA  E POI RITARDA PER UN
003 022 003          0       BREVE PERIODO DI TEMPO
003 023 333          IN      SOTTOPONE A TEST L'USCITA DEL
                             COMPARATORE
003 024 006          006     SE A = 1, È TROPPO ALTA; SE = 0 È TROPPO
                             BASSA
003 025 346          ANI
003 026 200          200
003 027 302          JNZ     SE È 1, L'APPROSSIMAZIONE È TROPPO ALTA
003 030 050          TOOHI
003 031 003          0
003 032 172  DIV2,   MOVAD   È TROPPO BASSA, AUMENTARE IL VALORE DI
                             APPROSSIMAZIONE
003 033 037          RAR     QUINDI SPOSTARE IL BIT DI TEST VERSO UNO
003 034 127          MOVDA   DEI BIT MENO SIGNIFICATIVI
003 035 173          MOVAE   ABBIAMO INOLTRE RUOTATO IL BYTE MENO
                             SIGNIFICATIVO
003 036 037          RAR     RUOTANDO UN EVENTUALE CARRY DA D
003 037 137          MOVEA   IN E.
003 040 031          DADD    ADDIZIONA D&E CON H&L, IL RISULTATO È IN
                             H&L
003 041 005          DCRB    TESTATI TUTTI I 10 BIT?
003 042 302          JNZ     NO, EFFETTUA IL TEST CON IL VALORE DI
                             APPROSSIMAZIONE SUCCESSIVA
003 043 010          DACOUT  METTENDOLO IN USCITA E VERIFICANDO
003 044 003          0       L'USCITA DEL COMPARATORE
003 045 303          JMP     SI, ABBIAMO TESTATO TUTTI I 10 BIT
003 046 000          ADC     QUINDI INCOMINCIA DI NUOVO
003 047 003          0
```

```
003 050 175     TOOHI,   MOVAL     L'APPROSSIMAZIONE ERA ECCESSIVA, QUINDI
003 051 223              SUBE      SOTTRAE IL BIT CORRENTE DA ENTRAMBI
003 052 157              MOVLA     H&L QUINDI RUOTA IL BIT DI TEST, LO SOMMA
003 053 174              MOVAH     CON IL DADE E RITENTA.
003 054 232              SBBD      (QUESTO IN CASO DI RIPORTO (BORROW))
003 055 147              MOVHA
003 056 303              JMP       ORA EFFETTUA IL TEST DEL BIT SUCCESSIVO
003 057 032              DIV2
003 060 003              0

                         SOTTOPROGRAMMA DI RITARDO

                         *003 200
003 200 365     DELAY,   PUSHPSW   SALVA I REGISTRI
003 201 325              PUSHD
003 202 021              LXID      CARICA I BYTE DI TEMPORIZZAZIONE NEI
                                   REGISTRI
003 203 046              046       D ED E
003 204 001              001
003 205 033     DEC,     DCXD      DECREMENTA LA COPPIA DI REGISTRI
003 206 172              MOVAD
003 207 263              ORAE      CONTROLLA CHE LA COPPIA DI REGISTRI SIA =
                                   000
003 210 302              JNZ       SE NON LO È, RITENTA
003 211 205              DEC
003 212 003              0
003 213 321              POPD      ESEGUITO, RIPRISTINA I REGISTRI
003 214 361              POPPSW
003 215 311              RET

                         *003 220
003 220 174     DISPLA,  MOVAH     PRENDE LA PARTE ALTA DEL BYTE DI DATI
003 221 323              OUT       E LO METTE IN USCITA SU UNA
003 222 000              000       DELLE PORTE CON LED
003 223 175              MOVAL     PRENDE LA PARTE BASSA DEL BYTE DI DATI E
003 224 323              OUT       LO METTE IN USCITA
003 225 002              002
003 226 315              CALL      DOPO AVER EMESSO H&L, RITARDA PER
003 227 200              DELAY     POCHI MSEC O SEC
003 230 003              0
003 231 311              RET
```

Passo 2

Cambiate il contenuto della locazione di memoria 003 204 in 200. Ciò prolungherà il tempo impiegato dal calcolatore per eseguire un sottoprogramma di ritardo.

Passo 3

Fate partire l'esecuzione del programma dall'indirizzo 003 000. Che cosa notate alle porte di uscita 000 e 002? Con il VOM collegato all'uscita del convertitore D/A, cosa notate sul quadrante mentre il programma è in esecuzione?

Se il programma viene eseguito nel modo corretto, dovreste vedere che alcuni LED

associati alle porte di uscita 000 e 002 lampeggiano ad una frequenza di circa un secondo. A seconda di come il vostro potenziometro è stato regolato, alcuni LED si accenderanno e rimarranno accesi, mentre altri si accenderanno e si spegneranno.

Dovreste inoltre notare che, su una particolare tensione, l'ago del VOM si mette in posizione di riposo.

Passo 4

Ruotate il regolatore del potenziometro fino a che non è ruotato completamente in senso anti-orario. Lasciate funzionare il calcolatore per almeno 10 secondi e annotate quello che potete osservare.

Ora ruotate il regolatore del potenziometro nel senso opposto (orario) completamente, e lasciate ancora funzionare il calcolatore per almeno 10 secondi prima di annotare le vostre osservazioni qui sotto:

Le vostre reali osservazioni dipendono dalla configurazione del vostro potenziometro, cioé dal fatto che la posizione oraria fornisca al comparatore il potenziale più alto o più basso.

Noi abbiamo scoperto che, nella posizione UNO, tutti tranne i due bit meno significativi in corrispondenza delle due porte di uscita erano a zero logico. L'ago del VOM è in posizione di riposo su una tensione che è vicina al potenziale di terra.

Sull'altra estremità, quasi tutti i LED rimangono accesi e l'ago del VOM sembrava riportarsi nella posizione di riposo quando si aveva un potenziale alto, circa + 5 volt.

Passo 5

Ruotate ancora il potenziometro nelle due posizioni estreme e osservate i vari stati dei LED quando il computer ritorna nella posizione di riposo alle tensioni alte e basse. Che cosa succede quando si ha in ingresso la tensione più alta?
E che cosa succede nel caso della tensione più bassa?

Noi abbiamo notato che i LED si illuminano, uno alla volta, dal bit più significativo a quello meno significativo in ciascun caso. Quando il potenziometro viene portato nel suo punto più alto, i LED si accendono e rimangono accesi. Quando invece il potenziometro viene regolato in modo da fornire al comparatore una bassa tensione, i LED si accendono e poi si spengono rimanendo spenti. Solo una o due delle posizioni corrispondenti ai bit meno significativi possono avere il loro LED acceso.

Passo 6

Regolate il potenziometro nel suo punto medio. Con il programma in esecuzione che cosa potete osservare in corrispondenza delle due porte di uscita?

Noi abbiamo notato che il computer eseguiva di nuovo la stessa sequenza di collaudo, cioé LED accesi e LED spenti. Ora alcuni LED rimanevano accesi mentre altri venivano spenti. Di nuovo, l'ago del VOM tornava nella posizione di riposo su una tensione compresa tra 0 e 5 volt verso il centro del quadrante.

Non abbiamo ancora fatto nessun commento sul visualizzatore collegato all'uscita del comparatore. Osservate questo indicatore e cercate di determinare il rapporto tra stato dell'uscita del comparatore e il fatto che i singoli LED che indicano se la posizione corrente del bit che viene collaudata, rimangano spenti o accesi. C'è una relazione secondo voi?

Noi abbiamo notato che esiste una precisa relazione tra lo stato dell'uscita del comparatore, indicato dal visualizzatore e gli stati acceso/spento del LED associato alla posizione corrente del bit che viene testato.

Se durante il test di una nuova posizione del bit il visualizzatore rimane acceso, il LED per quella specifica posizione si spegnerà e verrà testata la posizione bit successiva. Se, tuttavia, il visualizzatore rimane spento durante il test della posizione del bit, il LED associato con la posizione del bit rimarrà acceso.

Passo 7

Modificate il contenuto della locazione di memoria 003 204 da 100 a 020. Quindi fate partire di nuovo il programma dall'indirizzo 003 000. Che cosa succede ai LED in corrispondenza delle porte di uscita 000 e 002 adesso?

Benché la velocità delle conversioni sia maggiore, al termine dell'approssimazione successiva, si hanno accesi e spenti gli stessi LED. La velocità di conversione non modifica il valore finale digitalizzato di uno stesso valore di tensione in ingresso.

Passo 8

Modificate ora il contenuto della locazione di memoria 003 204 da 020 a 004, e fate partire il programma dall'indirizzo 003 000. Ciò produrrà un ulteriore aumento della velocità di conversione. Cosa notate, ora, nei LED associati alle porte di uscita 000 e 002?

Noi abbiamo notato che la conversione era così veloce che non eravamo neppure in grado di seguirla.

Se avete a disposizione un oscilloscopio, leggete attentamente i Passi 9 e 10, altrimenti passate oltre.

Passo 9

In questa fase proviamo a togliere il sottoprogramma di ritardo dalla sequenza di istruzioni eseguite dal programmma di conversione ad approssimazioni successive. Cambiate il contenuto della locazione 003 200 da 365 a 311. Con quali conseguenze? Collegate l'oscilloscopio all'uscita della tensione del convertitore D/A.

L'istruzione 311 che avete inserito nel vostro programma è un'istruzione di ritorno, RET. Ponendola all'inizio del sottoprogramma di DELAY, si ottiene che il sottoprogramma restituisce immediatamente il controllo al programma di approssimazione successiva.

Con l'oscilloscopio collegato all'uscita del convertitore D/A, dovreste essere in grado di osservare i vari salti di tensione come vengono messi in uscita dal convertitore D/A (sotto controllo di programma). Nello spazio che segue, schizzate un ciclo della traccia dell'oscilloscopio mentre le tensioni di prova tentano di portarsi alla tensione sconosciuta:

Il vostro schizzo dovrà essere simile al disegno nella Figura 2-6 nel Capitolo 2. Ricordate di testare sempre prima i valori di tensione più alti.

Passo 10

Collegate l'altro ingresso dell'oscilloscopio al punto di connessione tra l'ingresso del comparatore e il punto centrale del potenziometro. Ciò vi permetterà di esaminare la tensione sconosciuta che viene digitalizzata. Con i due guadagni di traccia regolati in modo uguale, posizionate la traccia della tensione sconosciuta sull'oscilloscopio in modo che coincida con l'ultimo valore di test della tensione di collaudo (minimo incremento). Ora modificate il valore impostato dal potenziometro. I tempi di conversione per le tensioni alte e quelle basse sono uguali?

La velocità di conversione è indipendente dal livello di tensione che è in fase di misura. Dovreste essere in grado di misurare sull'oscilloscopio dieci livelli di tensione e di test. Per eseguire la conversione vi occorrerà un convertitore A/D ad approssimazioni successive a N passi per ogni Nesimo passo. Noi abbiamo notate che, per un 8080 con un periodo di clock di 1,33 microsecondi, il computer richiede circa tre millisecondi per eseguire una conversione.

Passo 11

A questo punto apporterete al programma le seguenti modifiche, che permetteranno al programma di aggiornare la visualizzazione dei dati in corrispondenza delle porte di uscita 000 e 002 solo dopo che sarà stata completata la conversione a 10 bit.

a) Modifica:	003 045	da JMP	303	a JMP	303
	003 046	ADC	000	DISPLA	220
	003 047		003		003
b) Modifica:	003 020	da CALL	315	a NOP	000
	003 021	DISPLA	220	NOP	000
	003 022		003	NOP	000
c) Modifica:	003 231	da RET	311	a JMP	303
	003 232		——	ADC	000
	003 233		——		003

Fate partire il programma di conversione ad approssimazioni successive, iniziando dall'indirizzo 003 000. Vedrete solo il risultato di una conversione sui LED in corrispondenza delle porte di uscita 000 e 002.

Restituite i passi di programma elencati più sopra al loro contenuto originale prima di proseguire con il prossimo esperimento. È importante che ciò venga fatto, dato che parte del programma utilizzato in questo esperimento verrà recuperato e usato anche in quello che segue.

NON DISCONNETTERE L'INTERFACCIA O SPEGNERE LA MACCHINA. L'hardware e parte del programma di questo esperimento saranno utilizzati nell'esperimento che segue

ESPERIMENTO N. 5

UN PICCOLO SISTEMA DI ACQUISIZIONE DATI

Scopo

Scopo di questo esperimento è di utilizzare un convertitore D/A a 10 bit per la visualizzazione dei dati e contemporaneamente per effettuare conversioni ad approssimazioni successive.
Il programma usato per questo esperimento sarà in grado di acquisire 128 punti di dati a 10 bit, e di visualizzarli non appena sono stati acquisiti.

Commento

L'acquisizione di dati viene effettuata generalmente usando un convertitore A/D controllato da calcolatore, mentre la visualizzazione dei dati acquisiti avviene dopo che questi sono stati messi in uscita verso una telescrivente o un convertitore D/A. In questo esperimento, per entrambi gli scopi utilizzeremo un convertitore D/A a 10 bit. Il programma è stato scritto in modo da utilizzare l'uscita del convertitore D/A per mettere in uscita valori dei dati (verso un oscilloscopio) quando il converitore non è usato per generare le uscite necessarie alla conversione ad approssimazione successiva a controllo di programma. Per questo esperimento, l'uso di un oscilloscopio è indispensabile.

Per fornire un tempo di ritardo fisso si è ricorsi a un ciclo di visualizzazione, mentre, per contare il numero di volte che il computer deve eseguire il ciclo di visualizzazione, prima che un altro ingresso analogico venga digitalizzato e posto in memoria, viene utilizzata la coppia di registri. In questo modo il computer riserverà la maggior parte del tempo per visualizzare i dati. Il buffer di dati visualizzato ha una lunghezza di 256 locazioni di memoria, ogni punto è di 10 bit, in modo che il buffer di dati può contenere 128 valori di dati, per ciascuno dei quali sono fissate due locazioni di memoria.

Schema del circuito

Il circuito usato per questo esperimento è esattamente uguale a quello degli Esperimenti 3 e 4. Per i dettagli ad esso relativi vi rimandiamo all'Esperimento 3. L'interfaccia con il convertitore D/A a 10 bit utilizza il circuito integrato della Analog Devices AD7522, con l'aggiunta di alcuni componenti esterni. Nell'Esperimento 1, Passo 1, troverete i dettagli relativi a questo tipo di interfaccia. In mancanza di questo circuito, sarà utilizzabile qualunque altro convertitore D/A a 10 bit a doppia bufferizzazione.

Passo 1

La circuiteria usata nell'Esperimento 3 o nell'Esperimento 4 è utilizzata anche in questo. Nel caso in cui il circuito completo non sia già interfacciato con il vostro calcolatore, consultate l'Esperimento 1, Passo 1 e l'Esperimento 3 per le indicazioni necessarie alla realizzazione dell'interfaccia.

Alcuni passi di programma usati nell'esperimento con il convertitore A/D ad approssimazioni successive e a controllo di programma, relativi all'Esperimento 4, verranno utilizzati anche qui. Si tratta dei passi di programma localizzati agli indirizzi compresi tra l'indirizzo 003 000 e l'indirizzo 003 056, inclusi. Li troverete nel listing di programma per questo esperimento. Se avete già effettuato l'Esperimento 4 sarà bene ricontrollare questi passi. Se sono già nella memoria a lettura/scrittura del vostro calcolatore, vi faranno risparmiare un po' di tempo.

Se non disponete, invece, di questi passi di programma in memoria, introduceteli. NON tentate di far eseguire questa parte del programma.

QUESTO PROGRAMMA UTILIZZA LA TECNICA DELLE APPROSSIMAZIONI SUCCESSIVE PER DETERMINARE IL VALORE DIGITALE DI UNA TENSIONE ANALOGICA SCONOSCIUTA UTILIZZANDO UN CONVERTITORE D/A AD4522 E UN COMPARATORE

```
                           * 003 000
003 000 041   ADC,         LXIH      H & L CONTENGONO L'APPROSSIMAZIONE
                                     CORRENTE
003 001 000                000
003 002 002                002       CHE È XX XXX X10 00 000 000
003 003 021                LXID      D & E CONTENGONO IL BIT CORRENTE
003 004 000                000       CHE È SOMMATO O SOTTRATTO
003 005 002                002       DALLA APPROSSIMAZIONE CORRENTE
003 006 006                MVIB      B VIENE USATO COME CONTATORE DI BIT
003 007 012                012       012 OTTALE = 10 DECIMALE (CONVERTITORE A
                                     10 BIT)
003 010 175   DACOUT,      MOVAL     PRENDE GLI OTTO BIT DEI DATI MENO
                                     SIGNIFICATIVI
003 011 323                OUT       LI METTE IN USCITA VERSO IL CONVERTITORE
003 012 004                004
003 013 174                MOVAH     PRENDE I DUE BIT DI DATI PIU' SIGNIFICATIVI
003 014 323                OUT       LI METTE IN USCITA VERSO IL CONVERTITORE
003 015 005                005
003 016 323                OUT       CARICA IL REGISTRO PROVVISORIO DEL
                                     CONVERTITORE
003 017 003                003
003 020 315                CALL      VISUALIZZA L'APPROSSIMAZIONE CORRENTE
003 021 220                DISPLA    E POI RITARDA PER UN
003 022 003                0         BREVE PERIODO DI TEMPO
003 023 333                IN        TESTA L'USCITA DEL COMPARATORE. SE È
003 024 006                006       UN 1, È TROPPO ALTO, SE 0, È TROPPO BASSA
003 025 346                ANI
003 026 200                200
003 027 302                JNZ       È A 1, IL VALORE DI APPROSSIMAZIONE È
                                     TROPPO ALTO
003 030 050                TOOHI
003 031 003                0
003 032 172   DIV2,        MOVAD     È TROPPO BASSO, AUMENTA IL VALORE DI
                                     APPROSSIMAZIONE
003 033 037                RAR       QUINDI SPOSTA DI UNA POSIZIONE IL BIT DI
                                     COLLAUDO
003 034 127                MOVDA     VERSO I BIT MENO SIGNIFICATIVI
003 035 173                MOVAE     ABBIAMO INOLTRE RUOTATO IL BYTE
003 036 037                RAR       MENO SIGNIFICATIVO, SPOSTANDO UN
                                     EVENTUALE RIPORTO
003 037 137                MOVEA     DA D IN E
003 040 031                DADD      SOMMA D & E CON H & L, IL RISULTATO È IN
                                     H & L
003 041 005                DCRB      ESEGUITO IL TEST SU TUTTI I DIECI BIT?
003 042 302                JNZ       NO, TENTA CON IL VALORE
                                     DELL'APPROSSIMAZIONE SUCCESSIVA
003 043 010                DACOUT    METTENDOLO IN USCITA E TESTANDO
003 044 003                0         L'USCITA DEL COMPARATORE
003 045 303                JMP       SI, ABBIAMO PROVATO CON TUTTI I DIECI BIT
003 046 000                ADC       QUINDI REINIZIA L'APPROSSIMAZIONE
003 047 003                0
003 050 175   TOOHI,       MOVAL     L'APPROSSIMAZIONE ERA TROPPO ELEVATA,
                                     QUINDI
003 051 223                SUBE      SOTTRAE IL "BIT" CORRENTE SIA DA H CHE
```

```
003 052 157        MOVLA     DA L, QUINDI RUOTA IL BIT DI TEST, LO SOMMA
003 053 174        MOVAH     AL VALORE CORRENTE E PROVA DI NUOVO
003 054 232        SBBD      QUESTO È IL CASO DI UN RIPORTO
003 055 147        MOVHA
003 056 303        JMP       ORA PROVA CON IL SUCCESSIVO BIT
```

Passo 2

Effettuate le seguenti modifiche al programma di approssimazione successiva che è stato incluso nel Passo 1:

1. Togliere la chiamata al sottoprogramma DISPLA presente negli indirizzi 003 020, 003 021 e 003 022, sostituendo le istruzioni esistenti con istruzioni NOP (nessuna operazione) per tutti e tre i byte. Un'istruzione NOP è uguale a zero.
2. Sostituire il byte di istruzione JMP all'indirizzo 003 056 con un'istruzione di ritorno, RET = 311.

Quali saranno le conseguenze di queste modifiche sul programma?

Abbiamo tolto la chiamata al sottoprogramma DISPLA in modo che il risultato della conversione non venga più visualizzato nei LED in corrispondenza delle porte di uscita 000 e 002.
Ciò serve inoltre a togliere il sottoprogramma di ritardo, DELAY, dato che viene usato dal sottoprogramma DISPLA solamente.

Abbiamo inoltre posto un'istruzione di ritorno nella locazione di memoria usata in precedenza per contenere un'istruzione di salto. Normalmente, l'istruzione di salto fa si che il computer esegua continuamente il programma del convertitore A/D ad approssimazioni successive. L'uso di un'istruzione di ritorno al posto di un salto cambia la parte ADC del programma in un sottoprogramma ADC, per cui verrà eseguita solamente una conversione quando viene chiamato il sottoprogramma ADC.

Passo 3

Caricate i passi di programma di controllo e acquisizione dati che seguono nella memoria del calcolatore.
NOTA: Questo programma inizia dall'indirizzo 003 100. Evitate di caricarlo in cima al sottoprogramma ADC!

```
                             QUESTO È UN PROGRAMMA DI CONTROLLO E
                             VISUALIZZAZIONE DI ACQUISIZIONE DI DATI. IL
                             SUCCESSIVO PROGRAMMA DI APPROSSIMAZIONE USATO È
                             QUELLO DELL'ESPERIMENTO 4

                             * 003 100
003 100 061      START,      LXISP    DEFINISCE L'AREA DI STACK
003 101 377                  377
003 102 003                  003
```

```
003 103 041             LXIH    CARICA UN INDIRIZZO PUNTATORE
003 104 000             000
003 105 002             002
003 106 345   LOOP,     PUSHH   MEMORIZZA L'INDIRIZZO DI MEMORIA
003 107 021             LXID    CARICA I BYTE DI TEMPORIZZAZIONE
003 110 300             300
003 111 000             000
003 112 041   INIT,     LXIH    DEFINISCE L'INDIRIZZO DEL BUFFER DI
                                VISUALIZZAZIONE
003 113 000             000
003 114 002             002
003 115 174   DISPLA,   MOVAH   PRENDE GLI OTTO BIT MENO SIGNIFICATIVI DEL
                                DATO
003 116 323             OUT     LI METTE IN USCITA VERSO IL CONVERTITORE D/A
003 117 004             004
003 120 054             INRL    INCREMENTA L'INDIRIZZO DI MEMORIA
003 121 176             MOVAM   PRENDE I DUE BIT PIU' SIGNIFICATIVI
003 122 323             OUT     LI METTE IN USCITA VERSO IL CONVERTITORE D/A
003 123 005             005
003 124 323             OUT     GENERA L'IMPULSO PER CARICARE I 10 BIT SUL
                                CONVERTITORE D/A
003 125 003             003
003 126 054             INRL    INCREMENTA DI NUOVO L'INDIRIZZO DELLA
                                MEMORIA
003 127 302             JNZ     TERMINATO?
003 130 115             DISPLA  NO, SALTA AL SOTTOPROGRAMMA DISPLA PER
                                VISUALIZZARE
003 131 003             0       IL PUNTO SUCCESSIVO
003 132 033             DCXD    SI, DECREMENTA IL CONTATORE DI CICLI (BYTE DI
                                TEMPORIZZAZIONE)
003 133 172             MOVAD   TUTTI E 16 I BIT DEL CONTATORE SONO = 0?
003 134 263             ORAE
003 135 302   ALFA,     JNZ     NO, INIZIALIZZA PER UN'ALTRA
                                VISUALIZZAZIONE
003 136 112             INIT    DEL CONTENUTO DEL BUFFER
003 137 003             0
003 140 315             CALL    SI, DOPO L'ATTESA ESEGUE UNA CONVERSIONE
003 141 000             ADC     DIGITALE/ANALOGICA
003 142 003             0
003 143 353             XCHG    SPOSTA I DATI H & L A D & E
003 144 341             POPH    PRENDE L'INDIRIZZO DEL PUNTATORE DALLO
                                STACK
003 145 163             MOVME   MEMORIZZA GLI OTTO BIT MENO SIGNIFICATIVI
003 146 054             INRL
003 147 162             MOVMD   MEMORIZZA I DUE BIT PIU' SIGNIFICATIVI
003 150 054             INRL
003 151 302             JNZ     SONO STATI ACQUISITI TUTTI I 128 PUNTI?
003 152 106             LOOP    NO, TORNA INDIETRO ED ESEGUE UN'ALTRA
                                SERIE
003 153 003             0       DI VISUALIZZAZIONI DEL BUFFER
003 154 076             MVIA    SI, CARICA L'ACCUMULATORE CON IL VALORE 303
003 155 303             303
003 156 062             STA     LO MEMORIZZA ALL'INDIRIZZO ALFA
003 157 135             ALFA
003 160 003             0
003 161 303             JMP     ORA INIZIALIZZA E VISUALIZZA I DATI
003 162 112             INIT
003 163 003             0
```

La maggior parte del tempo del calcolatore viene destinata all'esecuzione della parte DISPLA del programma, fino all'istruzione JNZ, localizzata all'indirizzo 003 127. Quanti cicli di clock sono necessari al computer per eseguire questi passi, compresa l'istruzione JNZ? Vi rimandiamo sia al *Bugbook VI*, pag. 18-68 che al *Manuale dell'Utente del sistema Intel 8080* per avere queste informazioni.

Nell'esperimento fatto, abbiamo riscontrato che occorrono 73 cicli macchina. Per visualizzare un valore a 10 bit, a 1,33 microsecondi per ciclo (periodo di clock del nostro calcolatore) sono necessari circa 97 microsecondi. Se il periodo di clock del vostro computer è diverso dal nostro, calcolate il tempo necessario per visualizzare un dato con il vostro calcolatore e annotatelo qui sotto:

Passo 4

Utilizzando ii risultato ottenuto nel Passo 3, quanto tempo occorre per eseguire il loop DISPLA 128 volte e visualizzare tutti i valori dei dati almeno una volta?

A noi è risultato un tempo di (97 microsecondi) $\times$ (128 punti) $=$ 12,4 millisecondi. È evidente che se il periodo di clock del vostro calcolatore è diverso dal nostro, il tempo da voi calcolato sarà diverso.

Se la coppia di registri D è posta a un valore di 000 300, quante volte verrà eseguito il loop di visualizzazione, visualizzando gruppi di 128 punti?

Il ciclo DISPLA verrà eseguito 192 volte per ogni blocco di 128 punti a 10 bit. Quanto tempo impiegherà il calcolatore ad eseguire il loop DISPLA se non conosciamo il tempo in più necessario alle poche istruzioni che seguono l'istruzione JNZ DISPLA?

Il nostro computer ha richiesto circa 2,4 secondi. Quindi, i dati per essere acquisiti dal sottoprogramma ADC, il converitore A/D ad approssimazioni successive e controllo di programma, verranno campionati una volta ogni 2,4 secondi, aggiornando i dati memorizzati.

Passo 5

Collegate l'oscilloscopio all'uscita di tensione del convertitore D/A, V_{DAC} o DACOUT. Per la memorizzazione dati dovreste disporre della memoria a lettura/scrittura dall'indirizzo 002 000 all'indirizzo 002 377.

Fate partire il programma dall'indirizzo 003 100. E scrivete le vostre osservazioni qui di seguito:

Durante l'esecuzione del nostro esperimento abbiamo notato che venivano visualizzati sull'oscilloscopio dati casuali. Va ricordato che questo non è una visualizzazione XI, ma Y-t, dove l'asse del tempo o asse-t viene fornita dalla circuiteria del generatore della base-tempo dell'oscilloscopio.

Passo 6

Mentre il programma è in fase di esecuzione, ruotate lentamente il potenziometro fino al limite massimo di rotazione, cercando di impiegare dai 20 ai 30 secondi. Notate qualche effetto particolare sui dati visualizzati?

Noi abbiamo notato che sull'oscilloscopio venivano immessi e visualizzati nuovi dati la cui posizione verticale era direttamente proporzionale al movimento del potenziometro. Inoltre abbiamo osservato che non appena venivano acquisiti nuovi dati, il visore lampeggiava brevemente.

Continuate con l'acquisizione dei dati fino a che non li avrete acquisiti tutti e 128. Se occorre, regolate la base tempo dell'oscilloscopio in modo da poter ottenere la visualizzazione completa dei dati. Che cosa succede al termine di acquisizione dei dati?

Noi abbiamo osservato che il calcolatore continuava a visualizzare i 128 punti nel "file" dei dati, senza che venissero aggiunti altri punti.

Passo 7

Senza cambiare la regolazione del potenziomentro, o modificare l'interfaccia, fate partire nuovamente il programma dall'indirizzo 003 100. Si ripete la sequenza di acquisizione dei dati che avete notato al Passo 6?

Noi abbiamo notato solo la visualizzazione del file di dati costituito dai dati acquisiti durante la fase descritta al Passo 6. Siete in grado di capire il perché? Esaminate i passi di programma dall'indirizzo di memoria 003 154 all'indirizzo 003 163. Se avete bisogno di un suggerimento, controllate il contenuto della locazione 003 135. Che cosa è successo?

Per far sì che il programma si modifichi dopo aver acquisito 128 dati, è stato utilizzato un piccolo espediente. Come sappiamo, il programma contiene dei passi che servono a sostituire un jump incondizionato con un'istruzione condizionata, salta-se-non-è-zero (JNZ = 302) all'indirizzo simbolico ALFA.
Dopo un passo completo attraverso il prgramma di acquisizione dati ha luogo la modifica che permette solo la visualizzazione del gruppo di dati.

Passo 8

Sostituite il codice di istruzione ora memorizzato all'indirizzo simbolico ALFA con il codice operativo per l'istruzione JNZ e fate ripartire il programma dall'indirizzo 003 100. Che cosa accade ora?

Il programma viene eseguito come previsto, acquisendo nuovi dati e visualizzandoli. I dati acquisiti in precedenza, al Passo 6, sono ancora contenuti nella memoria del buffer per cui verranno visualizzati fino a quando tutti i vecchi dati verranno sostituiti da quelli nuovi.

Passo 9

L'espediente software che abbiamo notato al Passo 7 fornisce un elemento in grado di evitare che vengano acquisiti nuovi punti di dati una volta che è stato stabilito il file completo di 128 punti. Potreste aggiungere passi di programma per inizializzare l'istruzione JNZ in ALFA all'inizio del programma? Quali istruzioni aggiungereste? Cercate di aggiungerle in modo che non provochino errori o spostino istruzioni che fanno già parte del programma.

Per quanto riguarda il nostro esperimento abbiamo aggiunto i seguenti passi:

```
003 073   076   MVIA    /Carica 302 in A
003 074   302   302
003 075   062   STA     /Li memorizza in ALFA
003 076   135   ALFA
003 077   003   0
```

Se lo desiderate potete provarle. Queste istruzioni eliminano la necessità di reinizializzare l'istruzione JNZ come indicato dal Passo 8. Se aggiungete questi passi di programma fate attenzione ad iniziare il vostro nuovo programma dall'indirizzo 003 073, anche se potete aggiungere passi di programma che inizializzerebbero il file di dati con un valore diciamo 000.

L'hardware utilizzato in questo esperimento sarà utilizzato anche in quello che segue. Non togliete il collegamento se prevedete di eseguire il prossimo esperimento. Il programma non verrà invece più usato, perciò, se lo desiderate, potete spegnere la macchina.

ESPERIMENTO N. 6

MISURE DI SOGLIA DI UNA PORTA CON UN CONVERTITORE D/A

Scopo

Lo scopo di questo esperimento è di misurare le tensioni alle quali l'uscita di una porta standard della serie 7400 commuta da zero logico a uno logico e viceversa.

Commento

Le tensioni possono essere misurate indirettamente, senza l'uso di un convertitore A/D o altro dispositivo di misura. In questo esperimento, invece, per misurare le tensioni di soglia di una porta, abbiamo utilizzato un microcalcolatore e un convertitore a 10 bit. Più in particolare si potranno usare o l'SN74L00 o l'SN74LS00. Il convertitore D/A viene usato per generare una rampa da applicare agli ingressi della porta. L'uscita di questa porta è visualizzata per segnalare il cambio di stato. La tensione, alla quale questa variazione avviene, verrà annotata con il suo valore binario, quello cioé applicato al convertitore D/A a 10 bit.

Schema del circuito

Il convertitore D/A a 10 bit, usato nei precedenti esperimenti, verrà impiegato anche in questo che ci apprestiamo ad esaminare. Se il collegamento non è stato realizzato, vi rimandiamo all'Esperimento 1, Passo 1, per le informazioni relative.

Se avete appena completato gli Esperimenti 3, 4 o 5, avrete già a disposizione la porta di ingresso necessaria per questo esperimento. Se non l'avete a disposizione, vi aiuteremo a realizzarne una più avanti.

I singoli schemi verranno forniti nei vari passi a cui essi si riferiscono.

Passo 1

Con il convertitore D/A a 10 bit interfacciato al vostro calcolatore, caricate il programma seguente nella memoria del calcolatore, collegate un oscilloscopio o un VOM all'uscita del convertitore D/A, DACOUT o V_{DAC} e fate partire questo programma iniziando dall'indirizzo 003 000.

```
                    QUESTO È UN BREVE PROGRAMMA DI COLLAUDO PER
                    IL CONTROLLO DEL GUADAGNO E DELL'OFFSET

                    * 003 000
003 000 041         LXIH      PREDISPONE UN VALORE DA METTERE IN
                              USCITA
003 001 000         000       VERSO IL CONVERTITORE D/A
003 002 000         000
003 003 175         MOVAL
003 004 323         OUT
003 005 004         004
003 006 174         MOVAH
003 007 323         OUT
003 010 005         005
003 011 323         OUT       EMETTE L'IMPULSO DI MEMORIZZAZIONE
                              VERSO
003 012 003         003
003 013 166         HLT       STOP
```

Quando il programma è stato eseguito, quel sarà la tensione che compare all'uscita del convertitore D/A?

Noi abbiamo notato che l'uscita aveva un valore molto vicino a quello di terra (zero volt). Se state usando un convertitore D/A a 10 bit AD7522, come illustrato nell'Esperimento 1, o se state usando la Scheda LR-35, regolate il potenziometro OFFSET fino a quando l'uscita del convertitore D/A sarà il più possibile vicina a zero volt. Se invece state usando una interfaccia equivalente, potrà esserci un controllo di offset.
Se esiste, effettuate i necessari aggiustamenti in modo che l'uscita del convertitore sia di zero volt. Se invece questo controllo non esiste passate alla fase successiva.

Passo 2

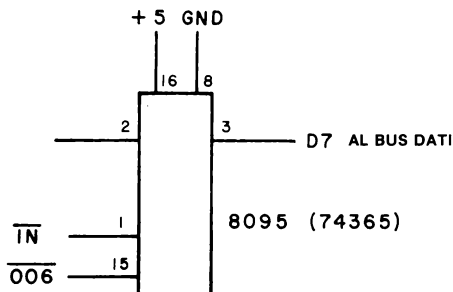
Sostituite il valore 377 per i valori presenti nelle locazioni di memoria con gli indirizzi 003 001 e 003 002. Fate ripartire il programma dall'indirizzo 003 000 e prendete nota dell'uscita di tensione del convertitore D/A nello spazio lasciato qui in basso:

In questa fase dell'esperimento noi abbiamo notato che l'uscita era di circa 4,6 volt. Se il vostro convertitore dispone di un controllo di GUADAGNO, regolatelo in modo che la tensione messa in uscita del convertitore D/A sia tra 5,0 e 5,1 volt. Se invece non ne dispone, passate alla fase seguente.

Passo 3

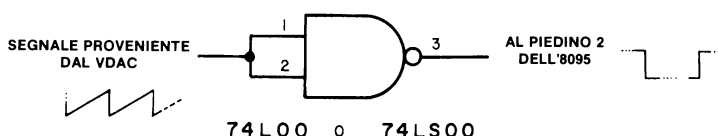
Ripetete le operazioni elencate al Passo 1 per assicurarvi che il convertitore D/A sia ancora azzerato e che metterà in uscita una tensione uguale a zero volt quando il codice di ingresso è zero.

Passo 4

Se avete appena completato gli Esperimenti 3, 4 o 5, togliete tutti i collegamenti relativi al comparatore LM311, ma lasciate la porta di ingresso three-state del DM8095 (SN74365) collegata al calcolatore.

Collegamenti necessari al DM8095 o all'SN74365 per configurarli come porta di ingresso three-state, IN 006.

Se sulla scheda di montaggio non avete una porta di ingresso collegate il circuito integrato DM8095 nel modo illustrato in questo passo. Assicuratevi che venga eseguito nel modo corretto il collegamento dei segnali di controllo IN e 006 al circuito integrato DM8095. L'alimentazione e la terra devono essere collegati come indicato dal disegno. Vi ricordiamo che l'SN74365 è equivalente al dispositivo DM8095.

Passo 5

Effettuate il collegamento della porta NAND come indicato dal disegno qui sotto, usandone una sola delle quattro disponibili. Vi consigliamo, per questa operazione, di usare l'SN74L00 o l'SN74LS00. Ricordate di collegare i piedini dell'alimentazione e della terra.

Passo 6

Immettete nella memoria del calcolatore il programma che segue. Questo programma genererà un'uscita di rampa lineare fornita agli ingressi della porta della serie SN7400.

```
                        * 003 000
003 000 041             LXIH     PREDISPONE IL VALORE INIZIALE DI RAMPA
003 001 000             000
003 002 000             000
003 003 175   RAMP,     MOVAL    METTE IL VALORE IN USCITA VERSO IL
                                 CONVERTITORE D/A
003 004 323             OUT
003 005 004             004
003 006 174             MOVAH
003 007 323             OUT
003 010 005             005
003 011 323             OUT      GENERA UN SEGNALE DI MEMORIZZAZIONE
                                 VERSO IL CONVERTITORE D/A
003 012 003             003
003 013 333             IN       INGRESSO DELLO STATO DELLA PORTA
003 014 006             006
003 015 346             ANI      MASCHERA GLI ALTRI BIT
003 016 200             200
003 017 312             JNZ      QUANDO COMMUTA È TERMINATO
003 020 026             DONE
003 021 003             0
003 022 043             INXH     AUMENTA IL VALORE DI RAMPA DI UNO
003 023 303             JMP      METTE IN USCITA IL NUOVO VALORE
003 024 003             RAMP
003 025 003             0
003 026 174   DONE,     MOVAH    PRENDE I DUE BYTE PIU' SIGNIFICATIVI
003 027 323             OUT      LI METTE IN USCITA VERSO LA PORTA 0
003 030 000             000
```

003 031 175	MOVAL	PRENDE GLI OTTO BIT PIU' SIGNIFICATIVI
003 032 323	OUT	E LI VISUALIZZA
003 033 002	002	
003 034 166	HLT	STOP

Passo 7

Fate partire il programma dall'indirizzo di memoria 003 000 e annotate, nello spazio lasciato qui di seguito, il valore a 10 bit ottenuto come risultato per ciascuna delle quattro prove. I quattro risultati andranno osservati sui LED collegati alle porte di uscita 000 e 002.

PRIMA PROVA ________________ SECONDA PROVA ________________

TERZA PROVA ________________ QUARTA PROVA ________________

Nel caso della porta SN74L00 abbiamo osservato i valori di 001, 021, 001 014, 001 013 e 001 014 (in codice ottale).

Queste letture possono essere trasformate in tensioni? Come?

Per un convertitore D/A a 10 bit ogni salto è di circa 5 millivolt. Le nostre letture erano di circa 1,34 volt.

Quali sono state le letture da voi misurate? Le tensioni reali che avete misurato dipenderanno dall'offset e dal guadagno del convertitore che state usando, dal rumore prodotto dall'alimentazione, ecc. Nei valori di tensione di soglia che siete in grado di misurare noterete delle differenze determinate dalla diversità esistente tra i diversi tipi di porte prodotte da fabbricanti diversi.

Con una rampa di tensione lineare ad andamento positivo avete misurato la tensione alla quale la porta NAND commuta la sua uscita da uno logico a zero logico.

Passo 8

In questa fase misureremo la tensione alla quale l'uscita della porta NAND commuta da zero logico ad uno logico. In questo caso quale sarà il tipo di rampa necessario?

Sarà necessaria una rampa con pendenza negativa. Quali modifiche di programma saranno necessarie per far sì che possa essere generata una rampa negativa con lo stesso programma base di cui già disponete?

Per generare una rampa negativa abbiamo apportato al programma le seguenti modifiche:

1) Abbiamo modificato i byte di dati caricati nei registri H e L con l'istruzione LXIH in modo da preparare nel registro L 377 e nel registro H 003. È nostra opinione che sarete perfettamente in grado di effettuare questa modifica senza bisogno di altro aiuto, basandovi esclusivamente sulla vostra esperienza di software.
2) Abbiamo modificato l'istruzione INXH con l'istruzione DCXX (053).

Sono, secondo voi, necessari altri cambiamenti?

SI. Deve essere cambiata l'istruzione che è stata usata per testare lo stato dell'uscita della porta, per segnalare il cambiamento avvenuto, al momento dell'avvio del programma, dello stato iniziale della porta.

Cambiate l'istruzione JZ con l'istruzione JNZ (302).

Passo 9

Fate partire il programma dall'indirizzo di memoria 003 000 e annotate di nuovo le vostre osservazioni nello spazio sottolineato qui sotto:

PRIMA PROVA ________________ SECONDA PROVA ________________

TERZA PROVA ________________ QUARTA PROVA ________________

Abbiamo osservato che le uscita di 001 051, 001 050 e 001 051 corrispondono alle tensioni di circa 1,5 volt. Questo naturalmente nel vostro sistema. La differenza tra le due tensioni di soglia che abbiamo misurato era di circa 160 millivolt.

Passo 10

Questo passo è facoltativo. Volete usare un'alimentazione variabile per fornire alimentazione alla porta NAND che viene sottoposta a collaudo? Vi ricordiamo che la tensione variabile deve essere mantenuta tra 4,75 e 7,0 volt per evitare che la porta possa essere danneggiata. Assicuratevi inoltre che tra l'alimentazione variabile e il sistema di elaborazione ci sia un buon collegamento di terra. Avete osservato degli effetti sulle tensioni di soglia al momento della variazione della tensione di alimentazione della porta? Le soglie sono rimaste le stesse per le diverse porte?

ESPERIMENTO N. 7

USO DI DUE CONVERTITORI D/A A OTTO BIT; IL MODULO BURR-BROWN MP-10

Scopo

Lo scopo di questo esperimento è di individuare dei campi di applicazione del doppio convertitore D/A MP-10 della Burr-Brown e in che modo può essere utilizzato sia come I/O con mappaggio della memoria sia come I/O tramite accumulatore.

Commento

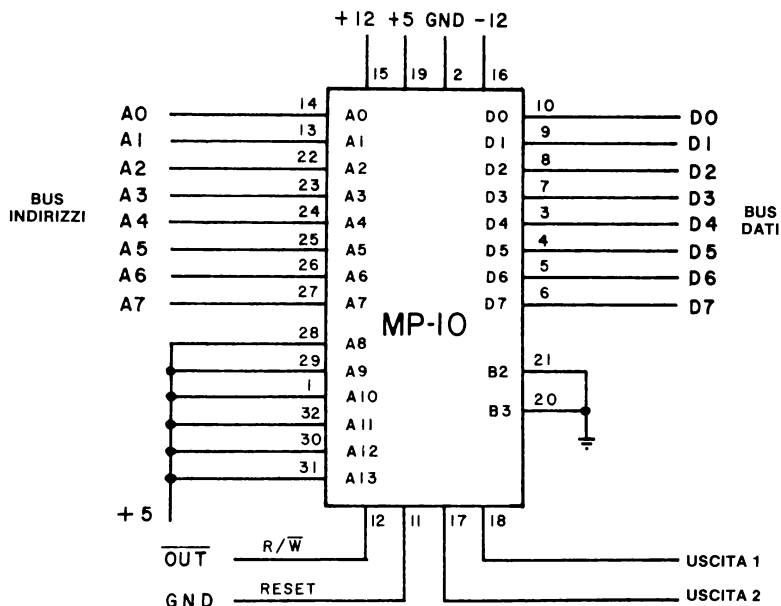
L'interfaccia con il convertitore MP-10 Burr-Brown incorpora tutta la logica necessaria per interfacciare due convertitori D/A a otto bit ad un microcalcolatore. I circuiti di conversione e decodifica necessari sono forniti nel modulo.

Il modulo MP-10 incorpora un circuito integrato di interfaccia periferica simile al dispositivo 8255 dell'Intel. Questo "chip" è sotto controllo del programmatore e può realmente essere programmato per modificare le sue funzioni interne tramite opportune istruzioni di programma. Nel modulo MP-10, viene usato un doppio latch bus-compatibile, che effettua la memorizzazione degli otto bit di ciascun convertitore D/A.

Quando si usano dispositivi di interfaccia programmabili, è necessaria fase di controllo del circuito, di solito all'inizio del programma. E questo è anche il caso dell'MP-10. Per l'esperimento che stiamo per illustrarvi è necessario un oscilloscopio.

Configurazione del pin di un modulo MP-10

Pin	Segnale		Segnale	Pin
1	A10		A11	32
2	Common		A13	31
3	D4		A12	30
4	D5		A 9	29
5	D6		A 8	28
6	D7		A 7	27
7	D3		A 6	26
8	D2		A 5	25
9	D1	MP10	A 4	24
10	D0		A 3	23
11	Reset		A 2	22
12	R/$\overline{W}$		B 2	21
13	A1		B 3	20
14	A0		+5V	19
15	+15V		Out 1	18
16	-15V		Out 2	17

Schema del circuito

Passo 1

Collegate il modulo MP-10 come indicato nel disegno qui sopra. Assicuratevi che il collegamento degli ingressi della alimentazione da + 12 e −12 sia stato effettuato correttamente. L'inversione della polarità può causare danni di una certa entità a questo tipo di modulo.

Passo 2

Una volta che il circuito è stato collegato, date alimentazione e interrompete temporaneamente il collegamento tra l'ingresso RESET dell'MP-10, il piedino 11 e il collegamento terra. Per ottenere il risultato migliore sollevate il collegamento fra i due punti, e provocando così l'azzeramento della logica interna dell'MP-10. Questo accorgimento deve essere applicato ogni volta che si dà alimentazione all'MP-10. Di solito questa funzione viene realizzata mediante un pulsante di reset.

Passo 3

Caricate il programma seguente nella memoria del calcolatore e fatelo partire dall'indirizzo 003 000.

	*003 000	
003 000 076	MVIA	PREDISPONE IN ACCUMULATORE IL VALORE PER L'INIZIALIZZAZIONE
003 001 200	200	

```
003 002 323           OUT     LO METTE IN USCITA VERSO I CONVERTITORI
                              D/A
003 003 363           363
003 004 043   LOOP,   INXH    INCREMENTA LA COPPIA DI REGISTRI
003 005 000           NOP
003 006 174           MOVAH   PRENDE I DATI DAL REGISTRO H
003 007 000           NOP
003 010 323           OUT     LI METTE IN USCITA VERSO IL CONVERTITORE
                              D/A
003 011 360           360
003 012 175           MOVAL   PRENDE I DATI DAL REGISTRO L
003 013 000           NOP
003 014 323           OUT     LI METTE IN USCITA VERSO IL CONVERTITORE
                              D/A
003 015 361           361
003 016 303           JMP
003 017 004           LOOP
003 020 003           000
```

Collegate un oscilloscopio alternativamente ad OUTPUT 1 e ad OUTPUT 2 dell'MP-10. Cosa notate?

Dovreste notare la presenza di due segnali varianti. Se ciò non succede, ricontrollate la vostra interfaccia e il programma contenuto in questa pagina. Per quanto riguarda l'esperimento eseguito da noi, abbiamo notato che l'MP-10 produceva due rampe negative, una con un periodo breve e l'altra con un periodo lungo.

Qual'è il campo di tensione delle due uscite analogiche?

Noi abbiamo osservato un'uscita compresa tra − 10 e +8 volt ogni volta che veniva applicata un'alimentazione tra + 12 e −12 volt. La Burr-Brown consiglia un'alimentazione di + 15 e −15 volt. Fornendo questo tipo di alimentazione abbiamo notato un campo di uscita di + 10 e −10 volt. *Potete usare un alimentatore separato per il ± 15 Volt preoccupandovi di effettuare un buon collegamento di massa tra il calcolatore e l'alimentatore.*

Passo 4

Dato che nel breve programma caricato nel calcolatore al Passo 3 viene usata un'istruzione di incremento e nelle uscite dell'MP-10 si nota una rampa negativa, cosa significa ciò relativamente sia ai dati analogici che a quelli digitali?

C'è una inversione dei dati, sia analogici che digitali.

L'interfaccia è predisposta come I/O con mappaggio della memoria o tramite accumultatore? Come fate a saperlo?

Viene usata la seconda soluzione. Per abilitare l'MP-10 viene usato il segnale $\overline{OUT}$ e per trasferire i dati dall'8080 ai convertitori D/A dell'MP-10 viene utilizzata un'istruzione OUT.

Passo 5

Inserire l'istruzione DCXH (053) al posto dell'istruzione INXH che già è nel programma alla locazione di memoria 003 004. Qual'è la conseguenza di questa sostituzione, quando l'esecuzione del programma riprende?

Le rampe hanno ora un andamento positivo. Ora inserite di nuovo l'istruzione INXH nella locazione di memoria 003 004. Siete in grado di suggerire altri tipi di modifiche di programma che potrebbero produrre lo stesso effetto di generare rampe ad andamento positivo? Ricordate che l'istruzione di decremento non può essere usata.

Noi abbiamo inserito due istruzioni di complemento del registro A (CMA = 057) al posto delle istruzioni non operative (NOP = 000) nelle locazioni 003 007 e 003 013.

Passo 6

Inserite le due istruzioni CMA nel programma come indicato al Passo 5. Ciò che avete fatto produce le uscite della rampa ad andamento positivo?

Noi abbiamo scoperto che l'effetto prodotto era proprio questo. Dovete sapere che esistono generalmente più possibili soluzioni per un dato problema, e che sono possibili sia soluzioni da programma che da hardware. Siete in grado di suggerire qualche soluzione da hardware per invertire le rampe generate nel programma originale presentato al Passo 3?

Invertite i collegamenti tra il bus dati e il modulo MP-10 o invertite il segnale di uscita analogico con un amplificatore operazionale.

Passo 7

Esaminate lo schema e la disposizione dei piedini del modulo MP-10. Noi abbiamo collegato i suoi ingressi di indirizzo da A13 a A8 a una tensione di + 5 Volt (1 logico).

Poiché l'MP-10 è stato studiato per essere impiegato prima di tutto come dispositivo di I/O in mappa di memoria, abbiamo collegato questi segnali di indirizzo in modo da simulare la presenza di bit di indirizzo alti come stato effettivo di funzionamento. Poi abbiamo collegato anche gli ingressi B2 e B3 a terra (0 logico). Questi sono gli indirizzi opzionali descritti nel Capitolo 1.

Con i collegamenti effettuati in questo modo, i codici a otto bit per indirizzare il dispositivo risultano i seguenti:

11110000	Convertitore D/A # 1
11110001	Convertitore D/A # 1
11110011	Registro di controllo interno

Nell'MP-10 il codice di indirizzo 11110010 non viene utilizzato.

Passo 8

Spegnete il calcolatore a le eventuali altre alimentazioni separate dell'MP-10 e togliete il collegamento tra l'indirizzo A13 (piedino 31) dell'MP-10 e i + 5 Volt. Collegate l'indirizzo A13 (piedino 31) dell'MP-10 con l'indirizzo A15 (piedino 36) dell'8080. Se non è già disponibile una possibilità di connessione si può realizzare con un piccolo cavallotto.

Poiché l'MP-10 ha bisogno che tutti gli ingressi degli indirizzi da A13 a A4 siano posti a zero logico per poter funzionare, che cosa implica questo nuovo collegamento all'indirizzo A15 dell'8080?

Questo collegamento significa che il modulo funzionerà solo quando uno degli indirizzi è presente, uno degli indirizzi del dispositivo elencati al Passo 7 in AND con l'uscita dell'indirizzo A15 dell'8080 posta a uno.

Passo 9

Staccate il collegamento tra il modulo MP-10 e il segnale $\overline{OUT}$ dell'8080. Ricollegate il piedino 12 dell'MP-10 e il segnale di scrittura in memoria dell'8080, $\overline{MW}$ o $\overline{MEMW}$. Questo collegamento e il collegamento tra l'MP-10 e l'indirizzo A15 dell'8080 riconfigurano l'MP-10 come dispositivo di I/O mappato in memoria.
ATTENZIONE: Questa configurazione non è assolutamente decodificata. L'MP-10 avrà molti indirizzi possibili nell'ambito dei 64K poiché non tutte le 16 linee di indirizzo dell'8080 sono state usate per decodificare gli indirizzi assegnati all'MP-10.

Passo 10

Caricate il programma che segue nella memoria del calcolatore. Si tratta di un programma generatore di rampa che utilizza dispositivi di I/O mappati in memoria, in questo caso l'MP-10. Ricordate di dare alimentazione al vostro sistema.

```
                        *003 000
003 000 041   START,    LXIH       CARICA NEI REGISTRI H&L L'INDIRIZZO DI
                                   MEMORIA
```

```
003 001 363            363     DEL REGISTRO DI CONTROLLO DEL
                               CONVERTITORE D/A
003 002 200            200
003 003 066            MVIM    CARICA LA PAROLA DI CONTROLLO
003 004 200            200     PAROLA DI CONTROLLO = 200
003 005 043   AGAIN,   INXH    INCREMENTA LA COPPIA DI REGISTRI H
003 006 042            SHLD    SALVA H&L AGLI INDIRIZZI
003 007 360            360     200 360 e 200 361, GLI INDIRIZZI
003 010 200            200     DEI CONVERTITORI D/A
003 011 303            JMP     RIPETE L'OPERAZIONE
003 012 005            AGAIN
003 013 003            0
```

Questo programma scrive la parola di controllo 200 nel Registro di Controllo Interno dell'MP-10 (indirizzo 200 363) per inizializzarlo. I due convertitori D/A a otto bit sono collocati agli indirizzi 200 360 e 200 361.

Una volta che il programma è stato caricato, togliere momentaneamente il collegamento a terra con l'ingresso di RESET dell'MP-10, piedino 11. (Possiamo scommettere che l'avevate dimenticato). Ora fate ripartire il programma. Che cosa siete in grado di notare?

Noi abbiamo notato che venivano di nuovo generate rampe ad andamento negativo.

Passo 11

Se desiderate usare l'interfaccia di I/O mappata in memoria per generare di nuovo rampe positive avrete bisogno di effettuare alcune modifiche sia di programma che di circuito. Potreste suggerire qualche tipico sottoprogramma da usare in questo caso? Cercate di non usare l'istruzione di decremento.

Nel nostro esperimento abbiamo potuto osservare che il programma che segue funziona molto bene. La coppia di registri D viene usata per conservare un conteggio ad incremento, mentre alla coppia di registri H vengono trasferiti i valori dei registri D ed E dopo essere stati complementati. Una volta che i dati si trovano nei registri H e L, vengono messi in uscita verso i convertitori D/A con l'istruzione SHLD.

```
                          *003 000
003 000 041               LXIH      AVETE GIA' VISTO QUESTA INIZIALIZZAZIONE
003 001 363               363       PRIMA
003 002 200               200
003 003 066               MVIM
003 004 200               200
003 005 023    RAMP,      INXD      INCREMENTA LA COPPIA DI REGISTRI D
003 006 172               MOVAD     PRENDE I DATI DAL REGISTRO D
003 007 057               CMA       LI COMPLEMENTA
003 010 147               MOVHA     LI MEMORIZZA NEL REGISTRO H
003 011 173               MOVAE     FA LA STESSA COSA PER IL REGISTRO E
003 012 057               CMA
003 013 157               MOVLA
003 014 042               SHLD      DI NUOVO, MEMORIZZA I REGISTRI H&L NEGLI
003 015 360               360       INDIRIZZI DI MEMORIA 200 360 e 200 361
003 016 200               200
003 017 303               JMP       CONTINUA
003 020 005               RAMP
003 021 003               0
```

Avrete probabilmente notato che si ha un piccolo vantaggio ad usare l'I/O mappato in memoria. Di fatto, se paragonata alla tecnica di I/O tramite accumulatore, la tecnica di I/O mappato in memoria richiede più passi di programma per eseguire le stesse operazioni. Molti considerano l'I/O con mappaggio della memoria come una panacea, in realtà non lo è e dovrebbe essere valutata più attentamente.

Passo 12

Spegnete il calcolatore. Se non prevedete di eseguire gli esperimenti che seguono potete smontare l'interfaccia; se, invece, pensate di proseguire con gli ultimi due esperimenti dovrete apportare alcune modifiche per ritrasformare l'interfaccia del convertitore D/A MP-10 nella configurazione di I/O tramite accumulatore.

Staccare il collegamento tra l'MP-10 e l'uscita $\overline{\text{MEMW}}$ dell'8080. Collegate il piedino 12 dell'MP-10 al segnale $\overline{\text{OUT}}$. Togliere il collegamento tra l'MP-10 e l'uscita dell'indirizzo A15 dell'8080 (8080 piedino 36) e collegare l'uscita dell'MP-10 (piedino 31) a + 5 volt.

ESPERIMENTO N.8

USO DI DUE CONVERTITORI D/A PER UNA VISUALIZZAZIONE SULL'ASSE Y IN FUNZIONE DI T

Scopo

Lo scopo di questo esperimento è di usare due convertitori a otto bit e il programma appropriato per fornire una visualizzazione di dati su un'Asse Y in funzione di T. Utilizzando un oscilloscopio sarà possibile visualizzare i diversi gruppi di dati.

Commento

In molti casi è necessario mettere in uscita un gruppo di dati come tensioni analogiche, che vengono visualizzate su un oscilloscopio o messe in uscita in direzione di un plotter per ottenere una registrazione permanente dei dati. In entrambi i casi l'uso dei due convertitori D/A semplifica enormemente la procedura di uscita.

Uno dei convertitori D/A viene utilizzato per mettere in uscita i dati analogici mentre l'altro viene usato per mettere in uscita una tensione continuamente in aumento in modo da poter disegnare mediante il plotter o visualizzare i punti con una linea continua da sinistra a destra. In questo esperimento si possono usare o il convertitore D/A MP-10 Burr-Brown o due convertitori D/A a otto bit indipendenti. Noi abbiamo riscontrato che il convertitore D/A a otto bit NE5018 della Signetics si presta molto bene ad essere usato per questo esperimento. Per quanto riguarda i dettagli di interfacciamento vi rimandiamo ai passi relativi a questo specifico problema.

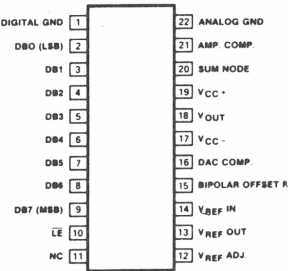
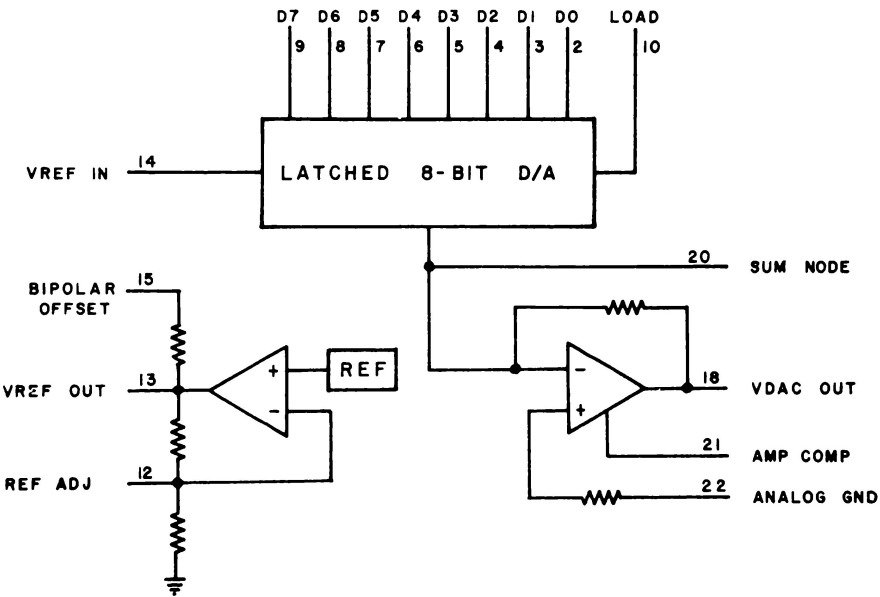
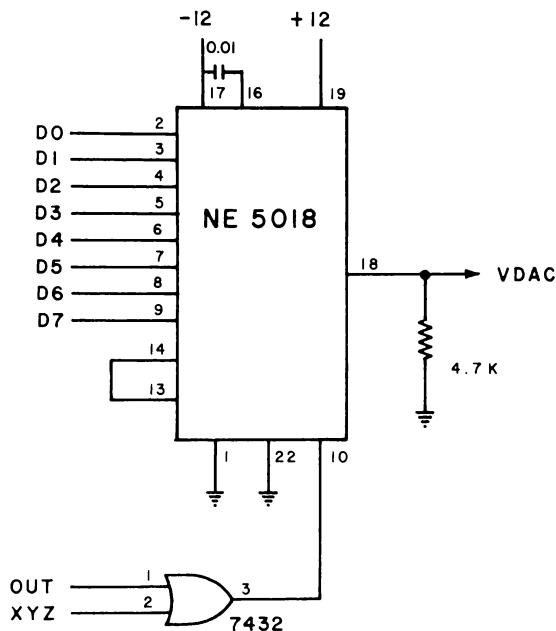
Schema del circuito

Lo schema reale del circuito varierà a seconda del tipo di convertitori che userete. Noi vi consigliamo di usare indifferentemente sia il modulo MP-10 Burr-Brown che i due dispositivi NE5018 della Signetics.

Passo 1

Se scegliete di lavorare con il modulo MP-10 Burr-Brown, vi rimandiamo all'Esperimento 7 per avere maggiori informazioni sul circuito di interfacciamento che è necessario e di proseguire con il Passo 3. Se invece usate il convertitore D/A a otto bit NE5018 della Signetics, continuate a partire dal Passo 2. Se, infine, avete deciso di utilizzare un altro tipo di convertitore D/A a otto bit, collegatelo, collaudatelo e infine proseguite con il Passo 3.

Passo 2

Il convertitore D/A a otto bit NE5018 della Signetics è uno strumento di facile impiego e poco costoso che può essere interfacciato ai microcalcolatori del tipo 8080. Il disegno qui in basso ne illustra lo schema a blocchi del circuito integrato, oltre alla disposizione dei piedini per il formato a 22 pin.

Noterete che l'NE5018 incorpora un convertitore D/A a otto bit con memorizzazione in ingresso, amplificatore in uscita e circuito di riferimento, il tutto sullo stesso circuito integrato. Vi sarà quindi molto facile eseguire questo interfacciamento, come si può notare dallo schema illustrato qui in basso.

Collegate due di queste interfacce al vostro calcolatore. Ricordate che dovrete fornire un decodificatore dell'indirizzo del dispositivo per gli indirizzi 360 e 361, ossia rispettivamente gli indirizzi da usare come ingresso "XYZ" per ciascuna delle interfacce. Per inizializzare l'MP-10 Burr-Brown, nel caso che lo si usi, si è ricorsi all'istruzione di uscita OUT 363. Ignorerete l'operazione di questo passo del programma se invece usate l'NE5018.

Passo 3

Per questo esperimento è richiesto l'uso di un oscilloscopio a doppia traccia con capacità di visualizzazione X-Y, o un oscilloscopio ad una traccia ma fornito della possibilità di applicare un ingresso esterno di base-tempo.

Collegate l'oscilloscopio in modo che l'uscita del convertitore D/A con il codice del dispositivo 361 fornisca la base tempo o l'ingresso dell'asse X e l'uscita del convertitore D/A con il codice del dispositivo 360 fornisca i dati per l'ingresso dell'asse Y.

Passo 4

Con i due convertitori D/A interfacciati con il calcolatore e collegati all'oscilloscopio, caricate il seguente programma nella memoria del calcolatore:

```
                             *003 000
003 000 061                  LXISP       DEFINISCE UN'AREA DI STACK
003 001 377                  377
003 002 003                  003
003 003 076                  MVIA        INIZIALIZZA IL MODULO MP-10
003 004 200                  200         SE VIENE USATO
003 005 323                  OUT
003 006 363                  363
003 007 041     NEW,         LXIH        PREDISPONE I PUNTATORI AL BUFFER DI DATI
003 010 000                  000
003 011 002                  002
003 012 176     LOOP,        MOVAM       PRENDE UN ELEMENTO DAL BUFFER
003 013 000                  NOP
003 014 323                  OUT         LA METTE IN USCITA VERSO IL CONVERTITORE
                                         D/A
003 015 360                  360
003 016 175                  MOVAL       PRENDE LA PARTE BASSA DELL'INDIRIZZO
                                         DELL'ELEMENTO
003 017 000                  NOP
003 020 323                  OUT         LA METTE IN USCITA VERSO L'ALTRO
                                         CONVERTITORE
003 021 361                  361
003 022 000                  NOP         AREA DI PROGRAMMA DISPONIBILE
003 023 000                  NOP
003 024 000                  NOP
003 025 054                  INRL        INCREMENTA LA PARTE BASSA
                                         DELL'INDIRIZZO
003 026 303                  JMP
003 027 012                  LOOP
003 030 003                  0
```

Questo programma mette in uscita 256 parole da 8 bit che sono memorizzate dalla locazione di memoria 002 000 alla locazione 002 377. I dati saranno rappresentati dalle tensioni in uscita del convertitore D/A che corrispondono a ingressi che vanno da 000 a 377 (inizio e fondo scala).

Passo 5

Con la base tempo interna dell'oscilloscopio collegata o opportunamente commutata per fornire la temporizzazione, fate partire il programma. Che cosa notate? Per quanto riguarda il nostro esperimento abbiamo dovuto regolare i controlli di trigger dell'oscilloscopio per mantenere fisse e costanti le informazioni visualizzate.

Sullo schermo dell'oscilloscopio abbiamo osservato un'immagine casuale ma ripetitiva. Assicuratevi di avere a disposizione della memoria a lettura/scrittura gli indirizzi dallo 002 000 allo 002 377.

Se la traccia risulta indistinta, potete aggiungere un sottoprogramma di ritardo in modo che ogni punto rimanga visualizzato per un periodo più lungo. A questo scopo risulta particolarmente adatto il sottoprogramma che segue e che abbiamo già visto nel corso degli esperimenti illustrati fino a qui.

SOTTOPROGRAMMA DI RITARDO

```
                          *003 200
003 200 365    DELAY,     PUSHPSW    SALVA I REGISTRI
003 201 325               PUSHD
003 202 021               LXID       PONE I BYTE DI TEMPORIZZAZIONE DEI
                                     REGISTRI
003 203 046               046        D ED E
003 204 001               001
003 205 033    DEC,       DCXD       DECREMENTA LA COPPIA DI REGISTRI
003 206 172               MOVAD
003 207 263               ORAE       CONTROLLA CHE LA COPPIA DI REGISTRI SIA =
                                     000
003 210 302               JNZ        SE NON È ZERO, RIPETE IL CONTROLLO
003 211 205               DEC
003 212 003               0
003 213 321               POPD       ESEGUITO, RIPRISTINA I REGISTRI.
003 214 361               POPPSW
003 215 311               RET
```

Ricordate di aggiungere la necessaria istruzione CALL e due byte di indirizzo al programma principale. Queste ultime possono essere introdotte al posto delle tre istruzioni NOP agli indirizzi 003 022, 003 023 e 003 024.

NOTA: Se state usando un convertitore D/A che inverte i dati analogici o digitali, al posto di due istruzioni NOP potrete usare, alle locazioni di memoria 003 013 e 003 017, le istruzioni di complemento (CMA = 057). Se preferite usare questo programma di ritardo, vi suggeriamo di iniziare con byte di temporizzazione uguali a 000 002.

A noi questo accorgimento è sembrato produrre un buon funzionamento.

Passo 6

È possibile esaminare i dati visualizzati e determinare l'inizio e la fine del gruppo di dati? Quale punto corrisponde ai valori memorizzati in 002 000?

Noi non siamo riusciti a stabilirlo. La visualizzazione era relativamente continua e non abbiamo avuto modo di distinguere i vari punti. Potete comunque sempre caricare alcuni valori, diciamo 000 e 377, nelle prime locazioni dell'area di memorizzazione della memoria. Queste verranno visualizzate per indicare l'inizio del gruppo dei dati.

Passo 7

Caricate il valore 377 in memoria dalla locazione 002 000 alla locazione 002 012, fate ripartire il programma e prendete nota di ciò che potete notare nello spazio qui sotto. È possibile distinguere questi punti dagli altri?

Sì, li abbiamo trovati nei dati che stavamo visualizzando. Si possono trovare inoltre una o più volte a seconda della base tempo dell'oscilloscopio e del trigger.

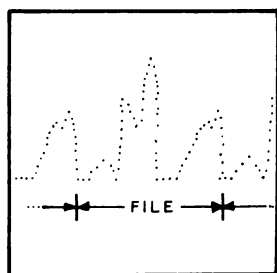
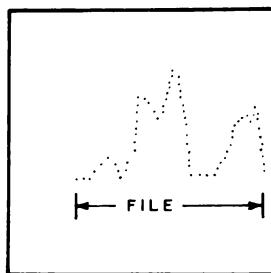
Passo 8

Abbiamo già detto che uno dei convertitori D/A può essere utilizzato per fornire all'oscilloscopio una base tempo esterna. Come avviene questo fatto? Siete in grado di esaminare il programma e stabilire una relazione tra i dati che sono visualizzati e il valore che viene messo in uscita verso l'altro convertitore D/A?

Un esame del programma indica che il convertitore D/A con il codice di dispositivo 361 mette in uscita una tensione che è proporzionale alla parte bassa dell'indirizzo che sarà costante per un dato punto. Ciò fornisce una rampa di tensione positiva che viene usata per generare il fascio elettronico che attraversa lo schermo nell'oscilloscopio da sinistra a destra.

Passo 9

Riconfigurate l'oscilloscopio per usare l'altro convertitore D/A (dispositivo 361) per fornire una base tempo esterna. Cosa notate?

Noi abbiamo notato che veniva visualizzato un insieme di punti di dati. Se il convertitore D/A che fornisce la rampa è collegato all'ingresso di base di tempo esterna, l'immagine visualizzata sarà costituita da una serie di dati, uno dopo l'altro come abbiamo già visto in precedenza. Se invece il convertitore D/A fornisce la rampa ad un ingresso sull'asse X, verrà visualizzata un'immagine singola dell'archivio di dati. Non ci sarà cioé nessuna visualizzazione in forma continua di un'immagine dopo un'altra. Ecco i due tipi di immagini visualizzate:

Rampa collegata alla base di tempi esterna

Rampa collegata all'asse X

L'oscilloscopio usato per questo esperimento è un modello Heath IO-4510, usato nel modo X-Y. L'immagine osservata era simile a quella indicata nella figura di destra. In

questo modo, è stato facile usare i controlli di guadagno degli assi X e Y per trasferire i dati in modo da riempire completamente lo spazio sullo schermo dell'oscilloscopio.

Passo 10

Immettete i seguenti valori nella memoria del calcolatore negli indirizzi indicati:

002 000 000	DATA,	000	002 012 040	040
002 001 001		001	002 013 050	050
002 002 002		002	002 014 060	060
002 003 003		003	002 015 070	070
002 004 004		004	002 016 120	120
002 005 005		005	002 017 140	140
002 006 006		006	002 020 160	160
002 007 007		007	002 021 200	200
002 010 020		020	002 022 220	220
002 011 030		030	002 023 240	240

Fate partire il calcolatore. Siete in grado di osservare i dati del gruppo dei 18 elementi che avete immesso? A cosa assomiglia?

Noi abbiamo notato una serie di tre rampe con pendenze in aumento. Ogni rampa risultava rappresentata da una serie di punti. Inoltre venivano visualizzati altri valori sempre contenuti nel gruppo di 256 punti.

Passo 11

Si potrebbe modificare il programma in modo che possano essere visualizzati meno di 256 punti? Siete in grado di suggerire qualche passo di programma adatto?

Nel nostro caso abbiamo approvato le seguenti modifiche. Queste modifiche visualizzano la parte bassa dell'indirizzo dell'elemento visualizzato. Quando viene raggiunto un indirizzo di limite (posto nel programma) il calcolatore visualizza di nuovo il gruppo cominciando dal primo dato.

	*003 025	
003 025 054	INRL	INCREMENTA LA PARTE BASSA DELL'INDIRIZZO
003 026 175	MOVAL	LA SPOSTA NEL REGISTRO A
003 027 376	CPI	LO METTE A CONFRONTO CON IL LIMITE SUPERIORE
003 030 000	000	MEMORIZZATO QUI
003 031 302	JNZ	SE NON È UGUALE CONTINUA A VISUALIZZARE
003 032 012	LOOP	GLI ELEMENTI DEL BUFFER
003 033 003	0	
003 034 303	JMP	SE VIENE RAGGIUNTO L'INDIRIZZO SUPERIORE
003 035 007	NEW	REINIZIALIZZA IL PUNTATORE ALL'INDIRIZZO
003 036 003	0	E VISUALIZZA DI NUOVO IL BUFFER

Caricate questi nuovi passi e fate partire il programma. Potreste anche provare i passi di programma che avete pensato come parte di questa fase dell'esperimento. In questo caso che cosa osservate?

Noi abbiamo notato che veniva di nuovo visualizzato il gruppo di dati, come prima. Ora, sostituite al valore 000 nella locazione di memoria 003 030 il valore 025, provocando così una modifica del limite superiore. Fate ripartire di nuovo il programma, e dite che cosa notate.

Vengono visualizzati solo i primi 26_8 dati. Per poter contenere più o meno valori nei dati visualizzati si può modificare il valore di confronto o la parte bassa dell'indirizzo dell'ultimo dato.

Sarebbe possibile mettere in uscita qualcosa di diverso da una linea continua o disegno a punti usando il metodo della visualizzazione X-T? Disegni per esempio?

No, poiché sull'asse del tempo, per ogni punto può essere visualizzato solo un punto, per cui l'uscita non può avere due valori analogici nello stesso istante.

Lasciate i convertitori collegati al vostro calcolatore, perché verranno utilizzati nel prossimo esperimento, mentre questa volta si può spegnere la macchina.

ESPERIMENTO N.9

L'USO DI DUE CONVERTITORI D/A PER UNA VISUALIZZAZIONE SUL PIANO X-Y

Scopo

Con questo esperimento indagheremo sull'uso di due convertitori D/A per l'uscita di dati sugli assi X-Y verso un visore per una visualizzazione di X in funzione di Y.

Commento

La tecnica della visualizzazione Y-T permette di visualizzare i dati che hanno solo un valore per ogni punto sull'asse X o asse del tempo, mentre la tecnica della visualizzazione di X in funzione di Y divide la superficie di visualizzazione in una matrice di punti, ciascuno dei quali può essere "indirizzato" fornendo una coordinata con un convertitore D/A e la seconda con l'altro convertitore D/A. Poiché useremo due convertitori D/A a otto bit, su ogni asse ci saranno 256 punti indirizzabili per un totale di 65,536 punti che possono essere indirizzati indipendentemente.

Per presentare poi la visualizzazione grafica dei dati, utilizzeremo un oscilloscopio.

Schema del circuito

Anche per questo esperimento useremo l'interfaccia già impiegata nell'Esperimento 8 a cui vi rimandiamo per maggiori dettagli.

Passo 1

Assicuratevi che i due convertitori D/A a otto bit, usati nell'Esperimento 8 siano ancora collegati con il vostro calcolatore. Se non lo fossero, consultate la parte dedicata ai dettagli sull'interfacciamento di quello stesso capitolo. Per il collaudo, nel nostro caso, abbiamo utilizzato un modulo MP-10 Burr-Brown e due circuiti integrati NE5018 Signetics.

Passo 2

Caricate il seguente programma nella memoria del calcolatore:

```
003 000 061              LXISP     DEFINISCE UN'AREA DI STACK
003 001 377              377
003 002 003              003
003 003 076              MVIA      INIZIALIZZA L'MP-10
003 004 200              200
003 005 323              OUT
003 006 363              363
003 007 041     NEW,     LXIH      CARICA L'INDIRIZZO DI INIZIO PER LE
                                   COORDINATE X
003 010 000              000
003 011 002              002
003 012 021              LXID      CARICA L'INDIRIZZO DI INIZIO PER LE
                                   COORDINATE Y
003 013 200              200
003 014 002              002
```

```
003 015 176   LOOP,   MOVAM   PRENDE UN PUNTO DELL'ASSE X
003 016 000           NOP
003 017 323           OUT     LO METTE IN USCITA
003 020 360           360
003 021 032           LDAXD   PRENDE UN DATO DELL'ASSE Y
003 022 000           NOP
003 023 323           OUT     METTE ANCHE QUESTO IN USCITA
003 024 361           361
003 025 000           NOP     SPAZIO DISPONIBILE
003 026 000           NOP
003 027 000           NOP
003 030 054           INRL    INCREMENTA IL PUNTATORE DELL'ASSE X
003 031 034           INRE    INCREMENTA ANCHE IL PUNTATORE
                              DELL'ASSE Y
003 032 302           JNZ     SE E NON È UGUALE A 0, VISUALIZZA IL
                              SUCCESSIVO
003 033 015           LOOP    PUNTO NEL BUFFER
003 034 003           0
003 035 303           JMP     SE IL BUFFER È VISUALIZZATO
                              COMPLETAMENTE RIPETE, COSI'
003 036 007           NEW     REINIZIALIZZA GLI INDIRIZZI DEI PUNTATORI.
003 037 003           0
```

Dopo aver caricato questo programma nella memoria del calcolatore, potete tornare indietro e controllarlo per correggere gli errori commessi durante il caricamento.

Passo 3

Collegate un oscilloscopio che abbia la capacità di visualizzare in modo che l'ingresso sull'asse X sia collegato al convertitore D/A che ha l'indirizzo del dispositivo 360. Come potete notare abbiamo di nuovo la stessa configurazione usata nell'Esperimento 8.

Fate partire il programma all'indirizzo 003 000 potreste aver bisogno di regolare il controllo di guadagno per ciascun ingresso in modo che i punti visualizzati riempiano lo schermo dell'oscilloscopio così come i vari comandi che regolano la posizione degli ingressi.

Annotate qui di seguito brevemente le vostre osservazioni:

Noi abbiamo notato che l'immagine visualizzata dei punti sullo schermo dell'oscilloscopio sono molto simili ai punti visualizzati negli esperimenti precedenti.

Tra i punti abbiamo notato anche delle macchie e delle striature, causate dalla temporizzazione del programma di visualizzazione. Il calcolatore impiega poco tempo per visualizzare i singoli punti per cui questi non risultano molto nitidi e brillanti. Poiché sono state inserite nel programma tre istruzioni di no-operation (NOP = 000), è possibile aggiungere un sottoprogramma di ritardo, che costringe il calcolatore a visualizzare ciascun punto per un periodo più lungo.

Per apportare questa modifica inserite, al posto delle tre istruzioni NOP la seguente istruzione di chiamata costituita da tre byte:

003 025	315	CHIAMATA
003 026	200	Parte bassa dell'indirizzo del sottoprogramma di ritardo.
003 003	003	Parte alta dell'indirizzo del sottoprogramma di ritardo.

Questo passo di programma esegue la chiamata al sottoprogramma di ritardo che avete usato nei precedenti esperimenti e che abbiamo inserito per maggiore chiarezza, anche in questo ultimo esperimento. Secondo noi, i byte di temporizzazione 000 e 002 nelle locazioni 003 204 e 003 205, rispettivamente, possono funzionare molto bene.

SOTTOPROGRAMMA DI RITARDO

```
                       *003 200
003 200 365   DELAY,   PUSHPSW   SALVA I REGISTRI
003 201 325            PUSHD
003 202 021            LXID      INSERISCE I BYTE DI TEMPORIZZAZIONE NEI
                                 REGISTRI
003 203 046            046       D ED E
003 204 001            001
003 205 033   DEC,     DCXD      DECREMENTA LA COPPIA DI REGISTRI
003 206 172            MOVAD
003 207 263            ORAE      CONTROLLA CHE LA COPPIA DI REGISTRI SIA
                                 UGUALE A 000
003 210 302            JNZ       SE NON È UGUALE A 0, RIPROVA
003 211 205            DEC
003 212 003            0
003 213 321            POPD      TERMINATO, RIPRISTINA I REGISTRI
003 214 361            POPPSW
003 215 311            RET
```

Passo 4

Caricate i seguenti dati nella memoria del calcolatore. Ogni dato ha un valore di coordinata X e un valore di coordinata Y.

Indirizzo	Dato	Indirizzo	Dato
002 000	000	002 200	000
002 001	040	002 201	000
002 002	140	002 202	000
002 003	200	002 203	377
002 004	300	002 204	377

Una volta che sono stati caricati i dati per l'asse X e quelli per l'asse Y per ciascuno dei cinque punti, fate ripartire il programma. Notate qualche differenza? Siete in grado di distinguere questi cinque punti dagli altri punti visualizzati?

Noi non abbiamo notato nessuna differenza e non siamo riusciti a distinguere questi cinque punti: sono troppi quelli da cui questi cinque debbono essere distinti.

Sarebbe meglio se il programma potesse essere modificato per visualizzare solo i punti che ci interessano. Si tratta di apportare lo stesso cambiamento già fatto nell'Esperimento 8.

Passo 5

Apportate al programma le seguenti modifiche, cambiando solo i passi elencati qui di seguito:

003 030 034	INRE	INCREMENTA L'INDIRIZZO DEL DATO DELL'ASSE Y
003 031 054	INRL	INCREMENTA IL PUNTATORE DELL'ASSE X
003 032 175	MOVAL	PRENDE IL CONTENUTO DEL REGISTRO L
003 033 376	CPI	LO CONFRONTA CON L'INDIRIZZO DI FINE
003 034 010	010	MEMORIZZATO QUI
003 035 302	JNZ	SE NON È UGUALE, VISUALIZZA
003 036 015	LOOP	IL PUNTO SUCCESSIVO
003 037 003	0	
003 040 303	JMP	SE È UGUALE, REINIZIALIZZA E
003 041 007	NEW	RIPARTE
003 042 003	0	

Dopo aver immesso e controllato questi nuovi passi di programma, fate ripartire il programma. Ora dov'este notare che vengono visualizzati solo pochi punti. Per ridurre le sbavature tra i punti provate ad utilizzare i byte di temporizzazione nel sottoprogramma di ritardo. Se state già usando il sottoprogramma di ritardo assicuratevi che l'istruzione di chiamata sia ancora nel programma principale.

Passo 6

Immettete i seguenti dati nella memoria del calcolatore:

Indirizzo	Dato	Indirizzo	Dato
002 000	000	002 200	000
002 001	040	002 201	040
002 002	100	002 202	100
002 003	040	002 203	140
002 004	020	002 204	200
002 005	010	002 205	240
002 006	000	002 206	300
002 007	000	002 207	340

Ora fate partire il programma. Cosa notate?

Noi abbiamo potuto notare una serie di otto punti che formavano un picco.

Passo 7

Caricate i dati seguenti nella memoria del calcolatore:

Indirizzo	Dato	Indirizzo	Dato
002 000	000	002 200	000
002 001	040	002 201	200
002 002	000	002 202	377
002 003	200	002 203	340
002 004	377	002 204	377
002 005	340	002 205	200
002 006	377	002 206	000
002 007	200	002 207	040

Fate di nuovo partire il programma e annotate le vostre osservazioni nello spazio qui sotto:

Noi abbiamo notato la visualizzazione di un quadrato con un punto per ogni lato.

Potrebbe essere visualizzato un vero quadrato in una visualizzazione del tipo Y in funzione di X?

No. È impossibile avere due punti con la stessa coordinata di "tempo". Si può tentare un'approssimazione, ma grafici più complessi non sono possibili.

Come si potrebbe aumentare il numero di punti che vengono visualizzati dal programma?

Potreste tornare indietro al "vecchio" programma descritto prima in questo stesso esperimento (per la visualizzazione di 128 punti) o modificare la parte del programma immesso al Passo 5 in modo che l'indirizzo di limite sia modificato per riflettere le vostre esigenze di visualizzazione. Il valore al limite viene memorizzato nella locazione di memoria con indirizzo 003 034. Questo programma ha la facoltà di visualizzare fino a 128 punti, ciascuno con due valori di coordinata a otto bit.

COMPONENTI NECESSARI PER GLI ESPERIMENTI ILLUSTRATI IN QUESTO VOLUME

Tutti gli esperimenti sono stati realizzati con l'impiego di un microcalcolatore 8080 che fornisce l'accesso ai segnali di controllo, del bus dati e del bus indirizzi per la scheda di montaggio. Nella preparazione e nel collaudo di questi esperimenti abbiamo utilizzato il sistema Mini-Micro Design della E&L Instruments a cui vi consigliamo di aggiungere almeno una scheda in più come la SK-50 o la SK-10.

Per gli esperimenti saranno necessari anche fili di diversa lunghezza del diametro di # 24 o # 26 e gli attrezzi di cui possono aver bisogno gli studenti, come pinze, un piccolo cacciavite per togliere i circuiti integrati dalla scheda di montaggio. Dovrete anche collegare i decodificatori o un dispositivo di visualizzazione, se non sono disponibili. Numerosi schemi per questi dispositivi sono contenuti nell'introduzione agli esperimenti contenuta in questo capitolo.

Sarà poi necessario avere a disposizione un voltmetro (che potrà anche essere del tipo meno costoso) e un oscilloscopio con due canali di ingresso o con un ingresso singolo e con una base-tempo esterna. Non sarà invece necessario avere un oscilloscopio per ciascuno studente dato che possono essere usati di volta in volta da tutti gli studenti.

Se decidete di costruire un circuito equivalente alla scheda da LR-35, vi consigliamo di costruirlo su un pezzo di scheda perforata o su una scheda di montaggio SK-50 o SK-10. Nell'Esperimento 1 troverete lo schema completo, per costruire il quale dovrete disporre dei seguenti componenti:

1	convertitore D/A a 10 bit AD7522, a doppia bufferizzazione (Analog Devices, Inc., Norwood, MA 02062)
1	circuito integrato 7402 comprendente quattro porte NOR
1	circuito integrato amplificatore operazionale 741
1	regolatore di tensione negativo a cinque volt 79L05
2	diodi 1N4001
1	potenziometro da 25 kohm a 10 giri
1	potenziometro da 5 kohm a 10 giri
1	resistore da 100 kohm, 1/4 di watt
1	resistore da 10 kohm, 1/4 di watt
1	diodo zener da 15 volt 1N965
2	condensatori al tantalio da 1 microfarad

Per gli esperimenti saranno necessari i seguenti componenti, oltre a quelli elencati più sopra per il convertitore D/A equivalente alla Scheda LR-35:

1	Scheda LR-35 o circuiteria equivalente[1]
1	buffer three-state DM8095 o SN74365
1	comparatore LM311
1	circuito a quattro porte NAND SN74L00 o SN74LS00
1	circuito a quattro porte OR SN7432
2	convertitore D/A a 8 bit "latched" NE5018[2] oppure
1	convertitore D/A a otto bit doppio MP-10[3] o circuiteria equivalente
1	outboard LR-6 con quattro visualizzatori o circuiti equivalenti. Vedere l'introduzione
2	resistori al carbonio 1/4 Watt, 1000 Ohm
2	resistori al carbonio 1/4 Watt, 4700 Ohm
1	potenziometro da 20 kohm, a 1 giro
2	condensatori da 0,01 microfarad ceramici
1	piccolo cacciavite per regolare il potenziometro

NOTE:

1. richiedere a: E & L Instruments, Inc.
61 First Street Derby, CT 06418

2. richiedere a: Signetics Corporation
811 East Arques Avenue
Sunnyvale, CA 94086

3. richiedere a: Burr-Brown Research
Corp. P.O. Box 11400
Tucson, AZ 85734

Per gli indirizzi consultare il produttore di ciascuna voce o un distributore locale.

APPENDICE 1

SPECIFICHE DEI CONVERTITORI A/D e D/A

Questa appendice presenta una nota di applicazione della National Semiconductor Corporation, *Specifyng A/D e D/A Converters* (AN-156).
Contiene molte utili definizioni e illustrazioni che integrano i concetti presentati nel libro.
La nota è stata stampata per gentile concessione della National Semiconductor Corp. Santa Clara California - 95051 -

SPECIFYING A/D AND D/A CONVERTERS

The specification or selection of analog-to-digital (A/D) or digital-to-analog (D/A) converters can be a chancey thing unless the specifications are understood by the person making the selection. Of course, you know you want an accurate converter of specific resolution; but how do you insure that you get what you want? For example, 12 switches, 12 arbitrarily valued resistors, and a reference will produce a 12-bit DAC exhibiting 12 quantum steps of output voltage. In all probability, the user wants something better than the expected performance of such a DAC. Specifying a 12-bit DAC or an ADC must be made with a full understanding of accuracy, linearity, differential linearity, monotonicity, scale, gain, offset, and hysteresis errors.

This note explains the meanings of and the relationships between the various specifications encountered in A/D and D/A converter descriptions. It is intended that the meanings be presented in the simplest and clearest practical terms. Included are transfer curves showing the several types of errors discussed. Timing and control signals and several binary codes are described as they relate to A/D and D/A converters.

MEANING OF PERFORMANCE SPECS

Resolution describes the smallest standard incremental change in output voltage of a DAC or the amount of input voltage change required to increment the output of an ADC between one code change and the next adjacent code change. A converter with n switches can resolve 1 part in 2^n. The least significant increment is then 2^{-n}, or one least significant bit (LSB). In contrast, the most significant bit (MSB) carries a weight of 2^{-1}. Resolution applies to DACs and ADCs, and may be expressed in percent of full scale or in binary bits. For example, an ADC with 12-bit resolution could resolve 1 part in 2^{12} (1 part in 4096) or 0.0245% of full scale. A converter with 10V full scale could resolve a 2.45mV input change. Likewise, a 12-bit DAC would exhibit an output voltage change of 0.0245% of full scale when the binary input code is incremented one binary bit (1 LSB). Resolution is a design parameter rather than a performance specification; it says nothing about accuracy or linearity.

Accuracy is sometimes considered to be a non-specific term when applied to D/A or A/D converters. A linearity spec is generally considered as more descriptive. An accuracy specification describes the worst case deviation of the DAC output voltage from a straight line drawn between zero and full scale; it includes all errors. A 12-bit DAC could not have a conversion accuracy better than ±½ LSB or ±1 part in 2^{12+1} (±0.0122% of full scale due to finite resolution). This would be the case in figure 1 if there were no errors. Actually, ±0.0122% FS represents a deviation from 100% accuracy; therefore accuracy should be specified as 99.9878%. However, convention would dictate 0.0122% as being an accuracy spec rather than an inaccuracy (tolerance or error) spec.

Accuracy as applied to an ADC would describe the difference between the actual input voltage and the full-scale weighted equivalent of the binary output code; included are quantizing and all other errors. If a 12-bit ADC is stated to be ±1 LSB accurate, this is equivalent to ±0.0245% or twice the minimum possible quantizing error of 0.0122%. An accuracy spec describes the maximum sum of all errors including quantizing error, but is rarely provided on data sheets as the several errors are listed separately.

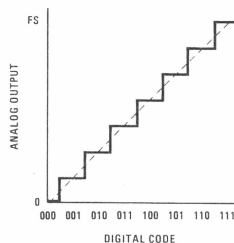



FIGURE 1. Linear DAC Transfer Curve Showing Minimum Resolution Error and Best Possible Accuracy

Quantizing Error is the maximum deviation from a straight line transfer function of a perfect ADC. As, by its very nature, an ADC quantizes the analog input into a finite number of output codes, only an infinite resolution ADC would exhibit zero quantizing error. A perfect ADC, suitably offset ½ LSB at zero scale as shown in figure 2, exhibits only ±½ LSB maximum output error. If not offset, the error will be $^{+1}_{-0}$ LSB as shown in figure 3. For example, a perfect 12-bit ADC will show a ±½ LSB error of ±0.0122% while the quantizing error of an 8-bit ADC is ±½ part in 2^8 or ±0.195% of full scale. Quantizing error is not strictly applicable to a DAC; the equivalent effect is more properly a resolution error.

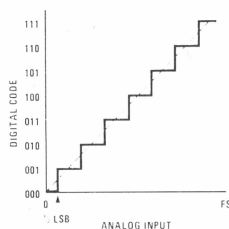



FIGURE 2. ADC Transfer Curve, ½ LSB Offset at Zero

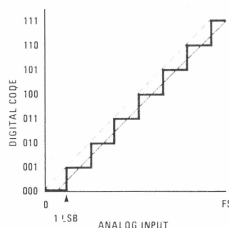



FIGURE 3. ADC Transfer Curve, No Offset

FIGURE 4. Linear, 1 LSB Scale Error

Gain Error is essentially the same as scale error for an ADC. In the case of a DAC with current and voltage mode outputs, the current output could be to scale while the voltage output could exhibit a gain error. The amplifier feedback resistors would be trimmed to correct the gain error.

Offset Error (zero error) is the output voltage of a DAC with zero code input, or it is the required mean value of input voltage of an ADC to set zero code out. (See figure 5.) Offset error is usually caused by amplifier or comparator input offset voltage or current; it can usually be trimmed to zero with an offset zero adjust potentiometer external to the DAC or ADC. Offset error may be expressed in % FS or in fractional LSB.

FIGURE 5. Linear, ½ LSB Offset Error

Scale Error (full scale error) is the departure from design output voltage of a DAC for a given input code, usually full-scale code. (See figure 4.) In an ADC it is the departure of actual input voltage from design input voltage for a full-scale output code. Scale errors can be caused by errors in reference voltage, ladder resistor values, or amplifier gain, *et. al.* (See **Temperature Coefficient.**) Scale errors may be corrected by adjusting output amplifier gain or reference voltage. If the transfer curve resembles that of figure 7, a scale adjustment at ¾ scale could improve the overall ± accuracy compared to an adjustment at full scale.

Hysteresis Error in an ADC causes the voltage at which a code transition occurs to be dependent upon the direction from which the transition is approached. This is usually caused by hysteresis in the comparator inside an ADC. Excessive hysteresis may be reduced by design; however, some slight hysteresis is inevitable and may be objectionable in converters if hysteresis approaches ½ LSB.

Linearity, or, more accurately, non-linearity specifications describe the departure from a linear transfer curve for either an ADC or a DAC. Linearity error does not include quantizing, zero, or scale errors. Thus, a specifi-

cation of ±½ LSB linearity implies error in addition to the inherent ±½ LSB quantizing or resolution error. In reference to figure 2, showing no errors other than quantizing error, a linearity error allows for one or more of the steps being greater or less than the ideal shown.

Figure 6 shows a 3-bit DAC transfer curve with no more than ±½ LSB non-linearity, yet one step shown is of zero amplitude. This is within the specification, as the maximum deviation from the ideal straight line is ±1 LSB (½ LSB resolution error plus ½ LSB non-linearity). With any linearity error, there is a differential non-linearity (see below). A ±½ LSB linearity spec guarantees monotonicity (see below) and ≤ ±1 LSB differential non-linearity (see below). In the example of figure 6, the code transition from 100 to 101 is the worst possible non-linearity, being the transition from 1 LSB high at code 100 to 1 LSB low at 110. Any fractional non-linearity beyond ±½ LSB will allow for a non-monotonic transfer curve. Figure 7 shows a typical non-linear curve; non-linearity is 1¼ LSB yet the curve is smooth and monotonic.

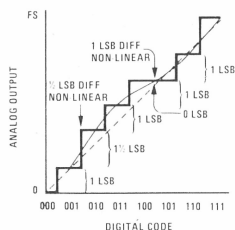



FIGURE 6. ±½ LSB Non-Linearity (Implies 1 LSB Possible Error), 1 LSB Differential Non-Linearity (Implies Monotonicity)

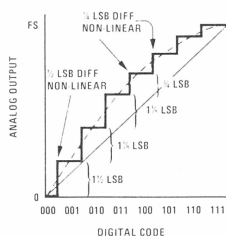



FIGURE 7. 1¼ LSB Non-Linear, ½ LSB Differential Non-Linearity

Linearity specs refer to either ADCs or to DACs, and do not include quantizing, gain, offset, or scale errors. Linearity errors are of prime importance along with differential linearity in either ADC or DAC specs, as all other errors (except quantizing, and temperature and long-term drifts) may be adjusted to zero. Linearity errors may be expressed in % FS or fractional LSB.

Differential Non-Linearity indicates the difference between actual analog voltage change and the ideal (1 LSB) voltage change at any code change of a DAC. For example, a DAC with a 1.5 LSB step at a code change would be said to exhibit ½ LSB differential non-linearity (see figures 6 and 7). Differential non-linearity may be expressed in fractional bits or in % FS.

Differential linearity specs are just as important as linearity specs because the apparent quality of a converter curve can be significantly affected by differential non-linearity even though the linearity spec is good. Figure 6 shows a curve with a ±½ LSB linearity and ±1 LSB differential non-linearity while figure 7 shows a curve with +1¼ LSB linearity and ±½ LSB differential non-linearity. In many user applications, the curve of figure 7 would be preferred over that of figure 6 because the curve is smoother. The differential non-linearity spec describes the smoothness of a curve; therefore it is of great importance to the user. A gross example of differential non-linearity is shown in figure 8 where the linearity spec is ±1 LSB and the differential linearity spec is ±2 LSB. The effect is to allow a transfer curve with grossly degraded resolution; the normal 8-step curve is reduced to 3 steps in figure 8. Similarly, a 16-step curve (4-bit converter) with only 2 LSB differential non-linearity could be reduced to 6 steps (a 2.6-bit converter?). The real message is, "Beware of the specs." Do not ignore or omit differential linearity characteristics on a converter unless the linearity spec is tight enough to guarantee the desired differential linearity. As this characteristic is impractical to measure on a production basis, it is rarely, if ever, specified, and linearity is the primary specified parameter. Differential non-linearity can always be as much as twice the non-linearity, but no more.

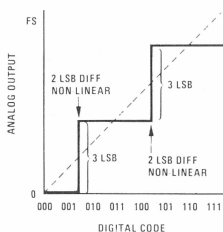



FIGURE 8. ±1 LSB Linear, ±2 LSB Differential Non-Linear

Monotonicity. A monotonic curve has no change in sign of the slope; thus all incremental elements of a monotonically increasing curve will have positive or zero, but never negative slope. The converse is true for decreasing curves. The transfer curve of a monotonic DAC will contain steps of only positive or zero height, and no negative steps. Thus a smooth line connecting all output voltage points will contain no peaks or dips. The transfer function of a monotonic ADC will provide no decreasing output code for increasing input voltage.

Figure 9 shows a non-monotonic DAC transfer curve. For the curve to be non-monotonic, the linearity error must exceed ±½ LSB no matter by how little. The greater the linearity error, the more significant the negative step might be. A non-monotonic curve may not be a special disadvantage in some systems; however, it is a disaster in closed-loop servo systems of any type (including a DAC-controlled ADC). A ±½ LSB maximum linearity spec on an n-bit converter guarantees monotonicity to n bits. A converter exhibiting more than ±½ LSB non-linearity may be monotonic, but is not necessarily monotonic. For example, a 12-bit DAC with ±½ bit linearity to 10 bits (not ±½ LSB) will be monotonic at 10 bits but may or may not be monotonic at 12 bits unless tested and guaranteed to be 12-bit monotonic.

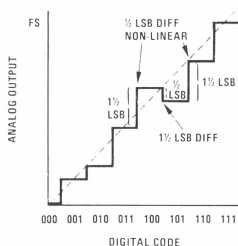



FIGURE 9. Non-Monotonic (Must be > ±½ LSB Non-Linear)

(a) Full-Scale Step

(b) 1 LSB Step

FIGURE 10. DAC Slew and Settling Time

Settling Time is the elapsed time after a code transition for DAC output to reach final value within specified limits, usually ±½ LSB. (See also **Conversion Rate** below.) Settling time is often listed along with a slew rate specification; if so, it may not include slew time. If no slew rate spec is included, the settling time spec must be expected to include slew time. Settling time is usually summed with slew time to obtain total elapsed time for the output to settle to final value. Figure 10 delineates that part of the total elapsed time which is considered to be slew and that part which is settling time. It is apparent from this figure that the total time is greater for a major than for a minor code change due to amplifier slew limitations, but settling time may also be different depending upon amplifier overload recovery characteristics.

Slew Rate is an inherent limitation of the output amplifier in a DAC which limits the rate of change of output voltage after code transitions. Slew rate is usually anywhere from 0.2 to several hundred volts/μs. Delay in reaching final value of DAC output voltage is the sum of slew time and settling time as shown in figure 10.

Overshoot and Glitches occur whenever a code transition occurs in a DAC. There are two causes. The current output of a DAC contains switching glitches due to possible asynchronous switching of the bit currents (expected to be worst at half-scale transition when all bits are switched). These glitches are normally of extremely short duration but could be of ½ scale amplitude. The current switching glitches are generally somewhat attenuated at the voltage output of the DAC because the output amplifier is unable to slew at a very high rate; they are, however, partially coupled around the amplifier via the amplifier feedback network and seen at the output. The output amplifier introduces overshoot and some non-critically damped ringing which may be minimized but not entirely eliminated except at the expense of slew rate and settling time.

Temperature Coefficient of the various components of a DAC or ADC can produce or increase any of the several errors as the operating temperature varies. Zero scale offset error can change due to the TC of the amplifier and comparator input offset voltages and currents. Scale error can occur due to shifts in the reference, changes in ladder resistance or non-compensating RC product shifts in dual-slope ADCs, changes in beta or reference current in current switches, changes in amplifier bias current, or drift in amplifier gain-set resistors. Linearity and monotonicity of the DAC can be affected by differential temperature drifts of the ladder resistors and switches. Overshoot, settling time, and slew rate can be affected by temperature due to internal change in amplifier gain and bandwidth. In short, every specification except resolution and quantizing error can be affected by temperature changes.

Long-Term Drift, due mainly to resistor and semiconductor aging can affect all those characteristics which temperature change can affect. Characteristics most commonly affected are linearity, monotonicity, scale, and offset. Scale change due to reference aging is usually the most important change.

Supply Rejection relates to the ability of a DAC or ADC to maintain scale, offset, TC, slew rate, and linearity when the supply voltage is varied. The reference must, of course, remain constant unless considering a multiplying DAC. Most affected are current sources (affecting linearity and scale) and amplifiers or comparators (affecting offset and slew rate). Supply rejection is usually specified only as a % FS change at or near full scale at 25°C.

Conversion Rate is the speed at which an ADC or DAC can make repetitive data conversions. It is affected by propagation delay in counting circuits, ladder switches and comparators; ladder RC and amplifier settling times; amplifier and comparator slew rates; and integrating time of dual-slope converters. Conversion rate is specified as a number of conversions per second, or conversion time is specified as a number of microseconds to complete one conversion (including the effects of settling time). Sometimes, conversion rate is specified for less than full resolution, thus showing a misleading (high) rate.

Clock Rate is the minimum or maximum pulse rate at which ADC counters may be driven. There is a fixed relationship between the minimum conversion rate and the clock rate depending upon the converter accuracy and type. All factors which affect conversion rate of an ADC limit the clock rate.

Input Impedance of an ADC describes the load placed on the analog source.

Output Drive Capability describes the digital load driving capability of an ADC or the analog load driving capacity of a DAC; it is usually given as a current level or a voltage output into a given load.

CODES

Several types of DAC input or ADC output codes are in common use. Each has its advantages depending upon the system interfacing the converter. Most codes are binary in form; each is described and compared below.

Natural Binary (or simply Binary) is the usual 2^n code with 2, 4, 8, 16, . . . , 2^n progression. An input or output high or "1" is considered a signal, whereas a "0" is considered an absence of signal. This is a positive true binary signal. Zero scale is then all "zeros" while full scale is all "ones."

Complementary Binary (or Inverted Binary) is the negative true binary system. It is identical to the binary code except that all binary bits are inverted. Thus, zero scale is all "ones" while full scale is all "zeros."

Binary Coded Decimal (BCD) is the representation of decimal numbers in binary form. It is useful in ADC systems intended to drive decimal displays. Its advantage over decimal is that only 4 lines are needed to represent 10 digits. The disadvantage of coding DACs or ADCs in BCD is that a full 4 bits could represent 16 digits while only 10 are represented in BCD. The full-scale resolution of a BCD coded system is less than that of a binary coded system. For example, a 12-bit BCD system has a resolution of only 1 part in 1000 compared to 1 part in 4096 for a binary system. This represents a loss in resolution of over 4:1.

Offset Binary is a natural binary code except that it is offset (usually ½ scale) in order to represent negative and positive values. Maximum negative scale is represented to be all "zeros" while maximum positive scale is represented as all "ones." Zero scale (actually center scale) is then represented as a leading "one" and all remaining "zeros." The comparison with binary is shown in figure 11.

Twos Complement Binary is an alternate and more widely used code to represent negative values. With this code, zero and positive values are represented as in natural binary while all negative values are represented in a twos complement form. That is, the twos complement of a number represents a negative value so that interface to a computer or microprocessor is simplified. The twos complement is formed by complementing each bit and then adding a 1; any overflow is neglected. The decimal number −8 is represented in twos complement as follows: start with binary code of decimal 8 (off scale for ± representation in 4 bits so not a valid code in the ± scale of 4 bits) which is 1000; complement it to 0111; add 0001 to get 1000. The comparison with offset binary is shown in figure 11. Note that the offset binary representation of the ± scale differs from the twos complement representation only in that the MSB is complemented. The conversion from offset binary to twos complement only requires that the MSB be inverted.

(a) Zero to + Full-Scale

(b) ± Full-Scale

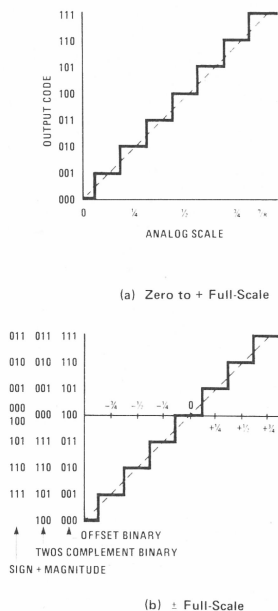



FIGURE 11. ADC Codes

Sign Plus Magnitude coding contains polarity information in the MSB (MSB = 1 indicates a negative sign); all other bits represent magnitude only. This code is compared to offset binary and twos complement in figure 11. Note that one code is used up in providing a double code for zero. Sign plus magnitude code is used in certain instrument and audio systems; its advantage is that only one bit need be changed for small scale changes in the vicinity of zero, and plus and minus scales are symmetrical. A DVM might be an example of its use.

CONTROL

Each ADC must accept and/or provide digital control signals telling it and/or the external system what to do and when to do it. Control signals should be compatible with one or more types of logic in common use. Control signal timing must be such that the converter or connected system will accept the signals. Common control signals are listed below.

Start Conversion (SC) is a digital signal to an ADC which initiates a single conversion cycle. Typically, an SC signal must be present at the fall (or rise) of the clock waveform to initiate the cycle. A DAC needs no SC signal; however, such could be provided to gate digital inputs to a DAC.

End of Conversion (EOC) is a digital signal from an ADC which informs the external system that the digital output data is valid. Typically, an EOC output can be connected to an SC input to cause the ADC to operate in continuous conversion mode. In non-continuous conversion systems, the SC signal is a command from the system to the ADC. A DAC does not supply an EOC signal.

Clock signals are required or must be generated within an ADC to control counting or successive approximation registers. The clock controls the conversion speed within the limitations of the ADC. DACs do not require clock signals.

CONCLUSION

Once the user has a working knowledge of DAC or ADC characteristics and specifications, he should be able to select a converter to suit a specific system need. The likelihood of overspecification, and therefore an unnecessarily high cost, is likewise reduced. The user will also be aware that specific parameters, test conditions, test circuits, and even definitions may vary from manufacturer to manufacturer. For practical production reasons, parameters may not be tested in the same manner for all converter types, even those supplied by the same manufacturer. Using information in this note, the user should, however, be able to sort out and understand those specifications (from any manufacturer) pertinent to his needs.

Manufactured under one or more of the following U.S. patents: 3083262, 3189758, 3231797, 3303356, 3317671, 3323071, 3381071, 3408542, 3421025, 3426423, 3440498, 3518750, 3519897, 3557431, 3560765, 3566218, 3571630, 3575609, 3579059, 3593069, 3597640, 3607469, 3617859, 3631312, 3633052, 3638131, 3648071, 3651565, 3693248.

National Semiconductor Corporation
2900 Semiconductor Drive, Santa Clara, California 95051, (408) 737-5000/TWX (910) 339-9240
National Semiconductor GmbH
808 Fuerstenfeldbruck, Industriestrasse 10, West Germany, Tele. (08141) 1371/Telex 05-27649
National Semiconductor (UK) Ltd.
Larkfield Industrial Estate, Greenock, Scotland, Tele. (0475) 33251/Telex 778-632

APPENDICE 2

DATA SHEETS DI UNA SELEZIONE DI PRODOTTI, CONVERTITORI

Questa appendice contiene alcuni data sheets di convertitori analogici/digitali, misuratori di pannello digitali e sistemi di acquisizione dati. La loro pubblicazione ha lo scopo di fornire esempi rappresentativi di quali informazioni sono contenute nei data sheets e fornisce informazioni addizionali circa alcuni dei dispositivi utilizzati e menzionati nel testo.
Alcuni dei data sheet sono stati pubblicati parzialmente per ottenere la massima concisione. È consigliabile, prima di acquistare un particolare dispositivo di prendere contatti diretti con i produttori per ottenere altra e più ampia documentazione sui vari prodotti.
L'appendice presenta i seguenti data sheets:

CMOS 10-Bit, Buffered Multiplying D/A Converter AD7522
Per gentile concessione della Analog Devices, Inc., Norwood, MA 02062

8-Bit μP-Compatible D-to-A Converter NE5018
Per gentile concessione della Signetics Corporation, Sunnyvale, CA 94086

MP-10, MP-11 Microprocessor Analog Output Components
Per gentile concessione della Burr-Brown Research Corporation, Tucson, AZ 85734

$3^1/_2$ Digit AC Line Powered DPM AD2009
Per gentile concessione della Analog Devices, Inc., Norwood, MA 02062

World's First Automatic Ranging Digital Panel meter Model DM-2000AR
Per gentile concessione della Datel Systems, Inc., Canton, MA 02021

Real-Time analog I/O Interface Model RTI-1200
Per gentile concessione della Analog Devices, Inc., Norwood, MA 02062

MP8600 Series, Burr- Brown Analog I/O Systems (per Microcomputers Intel)
Per gentile concessione della Burr-Brown Research Corporation, Tucson, AZ 85734

ADC0816/ADC0817 Single-Chip Data-Acquisition System
Per gentile concessione della National Semiconductor Corporation, Santa Clara, CA 95051

Esistono molti altri fabbricanti di moduli, allo scopo suggeriamo di consultare i nuovi prodotti presentati sulla stampa specializzata.
La più recente edizione del *"Gold Book"* di Electronic Design elenca più di 80 fabbricanti di convertitori analogici/digitali spaziando dai circuiti integrati ai sistemi completi.

ANALOG DEVICES

CMOS 10-Bit, Buffered Multiplying D/A Converter

AD7522

FEATURES

10-Bit Resolution
8, 9, & 10-Bit Linearity
Microprocessor Compatible
Double Buffered Inputs
Serial or Parallel Loading
DTL/TTL/CMOS Direct Interface
Nonlinearity Tempco: 2ppM of FSR/°C
Gain Tempco: 10ppM of FSR/°C
Very Low Power Dissipation
Very Low Feedthrough

GENERAL DESCRIPTION

The AD7522 is a monolithic CMOS 10-bit multiplying D/A converter, with an input buffer and a holding register, allowing direct interface with microprocessors. Most applications require the addition of only an operational amplifier and a reference voltage.

The key to easy interface to a data bus is the AD7522's ability to load the input buffer in two bytes (an 8-bit and a 2-bit byte), and subsequently move this data to a holding register, where the digital word is converted into an analog current or voltage (with external operational amplifier). The input loading of either 8 or 10 bits can be done in a parallel or serial mode.

The AD7522 is packaged in a 28-pin DIP, and operates with a +15V main supply at 2mA max, and a logic supply of +5V for TTL interface, or +10 to +15V for CMOS interface.

A thin film on high density CMOS process, using silicon nitride passivation, ensures high reliability and excellent stability.

FUNCTIONAL DIAGRAM

ORDERING INFORMATION

Nonlinearity	Temperature Range		
	0°C to +70°C	-25°C to +85°C	-55°C to +125°C
0.2% (8-Bit)	AD7522JN	AD7522JD	AD7522SD
0.1% (9-Bit)	AD7522KN	AD7522KD	AD7522TD
0.05% (10-Bit)	AD7522LN	AD7522LD	AD7522UD

PIN CONFIGURATION

PACKAGE IDENTIFICATION

Suffix "D": Ceramic DIP Package
Suffix "N": Plastic DIP Package

Route 1 Industrial Park; P.O.Box 280; Norwood,Mass. 02062
Tel: 617/329-4700 TWX: 710/394-6577
West Coast 213/595-1783
Mid-West 312/894-3300
Texas 214/231-5094

SPECIFICATIONS (VDD = +15V, VCC = +5V, VREF = +10V, TA = +25 C unless otherwise noted)

PARAMETER[1]	VERSION	TA = 25°C MIN	TA = 25°C TYP	TA = 25°C MAX	OVER SPECIFIED TEMP. RANGE MIN	OVER SPECIFIED TEMP. RANGE MAX	UNITS	TEST CONDITIONS
STATIC ACCURACY								
Resolution	All	10			10		Bits	SC8 = "1"
Nonlinearity	J			±0.2			% FSR	−10V ≤ VREF ≤ +10V
	S			±0.2		±0.2	% FSR	
	K			±0.1			% FSR	
	T			±0.1		±0.1	% FSR	
	L			±0.05			% FSR	
	U			±0.05		±0.05	% FSR	
Nonlinearity Tempco[2]	J, K, L		±1			±2	ppm FSR/°C	
	S, T, U					±2	ppm FSR/°C	
Gain Error	J, K, L		±0.3				% Reading	
Gain Error Tempco[2]	J, K, L		±5			±10	ppm of Reading/°C	
	S, T, U					±10	ppm of Reading/°C	
Output Leakage Current at IOUT1 or IOUT2	All					200	nA	IOUT1: DB0 through DB9 = 0 IOUT2: DB0 through DB9 = 1
Power Supply Rejection	J, K, L		50				ppm of Reading/%	
AC ACCURACY								
Feedthrough Error[2]	All		1	10			mV p-p	VREF = 20V p-p; 10 kHz
Output Current Settling Time	J, K, L		500				ns	To 0.05% of FSR for a FSR Step. HBS and LBS Low to High LDAC = 1
REFERENCE INPUT								
Input Resistance	All	5		20			kΩ	
ANALOG OUTPUT								
Output Capacitance								
COUT1	J, K, L		120				pF	All Data Inputs High
COUT2	J, K, L		40				pF	
COUT1	J, K, L		40				pF	All Data Inputs Low
COUT2	J, K, L		120				pF	
DIGITAL INPUTS								
Low State Threshold	All			0.8		0.8	V	VCC = +5V
	All			1.5		1.5	V	VCC = +15V
High State Threshold	All	2.4			2.4		V	VCC = +5V
	All	13.5			13.5		V	VCC = +15V
Input Current	J, K, L		1				µA	
LDAC Pulse Width[2]	All	500			500		ns	LDAC: 0 to +3V
HBS, LBS Pulse Width[2]	All	500			500		ns	HBS, LBS: 0 to +3V
Serial Clock Frequency[2]	All			1		1	MHz	
HBS, LBS Data Set Up[3]	All	250			250		ns	
Data Hold Time[4]	All	500	200		500		ns	

Notes
[1] Specifications subject to change without notice.
[2] Guaranteed by design. Not tested.
[3] Data setup time is the minimum amount of time required for DB0 - DB9 to be stable prior to strobing HBS, LBS.
[4] Data hold time is the minimum amount of time required for DB0 - DB9 to be stable after strobing HBS, LBS.

SPECIFICATIONS (continued)

PARAMETER[1]	VERSION	TA = +25°C			OVER SPECIFIED TEMP. RANGE		UNITS	TEST CONDITIONS
		MIN	TYP	MAX	MIN	MAX		
POWER REQUIREMENTS								
IDD	All			2			mA	In Quiescent State
ICC	All			2			mA	
PRICE (100-499)								
8 Bit	JN	$ 15.00						
	JD	21.00						
	SD	41.00						
9 Bit	KN	$ 17.00						
	KD	25.00						
	TD	51.00						
10 Bit	LN	$ 25.00						
	LD	37.00						
	UD	99.00						

ABSOLUTE MAXIMUM RATINGS

VREF to GND . . . . ±25V
VDD to GND . . . . +17V
VCC to GND . . . . +17V
VCC to VDD . . . . +0.4V
IOUT1, IOUT2 . . . . ±5 mA
Operating Temperature
JN, KN, LN versions . . . . 0°C to +70°C
JD, KD, LD versions . . . . −25°C to +85°C
SD, TD, UD versions . . . . −55°C to +125°C

Storage Temperature . . . . −65°C to +150°C
Power Dissipation (Package)
Up to +50°C:
Plastic (Suffix N) . . . . 1200 mW
Ceramic (Suffix D) . . . . 1000 mW
Derate Above +50°C by
Plastic (Suffix N) . . . . 12 mW/°C
Ceramic (Suffix D) . . . . 10 mW/°C
Digital Input Voltage Range . . . . . . VDD to GND

CAUTION:

1. Do not apply voltages higher than VCC to SRO.
2. Do not apply voltages higher than VDD or less than GND to any other input/output terminal except VREF, RFB1 or RFB2.
3. The digital control inputs are zener protected, however permanent damage may occur on unconnected units under high energy electrostatic fields. Keep unused units in conductive foam at all times.
4. VCC should never exceed VDD by more than 0.4V, especially during power ON or OFF sequencing.

Terminology

RESOLUTION

Value of the LSB. For example, a unipolar n-bit converter has a resolution of (2^{-n}) (VREF). A bipolar n-bit converter has a resolution of $[2^{-(n-1)}]$ [VREF]. Resolution in no way implies linearity.

NONLINEARITY

Error contributed by deviation of the DAC transfer function from a best straight line function. For a multiplying DAC, the nonlinearity should be independent of the sign or magnitude of VREF. Nonlinearity is normally expressed as a percentage of full scale range (% FSR).

GAIN

The "gain" of a converter is that analog scale factor setting that establishes the nominal conversion relationship, e.g., 10V full scale. It is a linear error which can be externally adjusted. (See gain adjustment on page 6.)

OUTPUT LEAKAGE CURRENT

Current which appears on the OUT1 terminal when the DAC register is loaded with all "0's" or on the OUT2 terminal when the DAC register is loaded with all "1's."

Pin Function Description

PIN	MNEMONIC	DESCRIPTION
1	VDD	+15V (nominal) Main Supply.
2	LDTR	R-2R Ladder Termination Resistor. Normally grounded for unipolar operation or terminated at IOUT2 for bipolar operation.
3	VREF	Reference Voltage Input. Since the AD7522 is a multiplying DAC, VREF may vary over the range of ±10V.
4	RFB2	Rfeedback ÷ 2; gives full scale equal to VREF/2.
5	RFB1	Rfeedback, used for normal unity gain (at full scale) D/A conversion.
6	IOUT1	DAC Current OUT-1 Bus. Normally terminated at virtual ground of output amplifier.
7	IOUT2	DAC Current OUT-2 Bus, terminated at ground for unipolar operation, or virtual ground of op amp for bipolar operation.
8	AGND	Analog Ground. Back gate of DAC N-channel SPDT current steering switches.
9	SRO	Serial Output. An auxiliary output for recovering data in the input buffer.
10	DB9 (Note 1)	Data Bit 9. Most significant parallel data input.
11	DB8 (Note 1)	Data Bit 8.
12	DB7 (Note 1)	Data Bit 7.
13	DB6 (Note 1)	Data Bit 6.
14	DB5 (Note 1)	Data Bit 5.
15	DB4 (Note 1)	Data Bit 4.
16	DB3 (Note 1)	Data Bit 3.
17	DB2 (Note 1)	Data Bit 2.
18	DB1 (Note 1)	Data Bit 1.
19	DB0 (Note 1)	Data Bit 0. Least significant parallel data input.
20	$\overline{SC8}$	8-Bit Short Cycle Control. When in serial mode, if $\overline{SC8}$ is held to logic "0," the two least significant input latches in the input buffer are bypassed to provide proper serial loading of 8-bit serial words. If $\overline{SC8}$ is held to logic "1," the AD7522 will accept a 10-bit serial word. Data bits 0(LSB) and DB1 are in a parallel load mode when $\overline{SC8}$ = 0, and should be tied to a logic low state to prevent false data from being loaded.
21	SPC	Serial/Parallel Control. If SPC is a logic "0," the AD7522 will load parallel data appearing on DB0 through DB9 into the input buffer when the appropriate strobe inputs are exercised (see HBS and LBS). If SPC is a logic "1," the AD7522 will load serial data appearing on Pin 26 into the input buffers. Each serial data bit must be "strobed" into the buffer with the HBS and LBS.
22	LDAC	Load DAC: When LDAC is a logic "0," the AD7522 is in the "hold" mode, and digital activity in the input buffer is locked out. When LDAC is a logic "1," the AD7522 is in the "load" mode, and data in the input buffer loads the DAC register.
23	NC	No Connection.
24	LBS	Low Byte Strobe. When in "parallel load" mode (SPC = 0), parallel data appearing on the DB0 (LSB) through DB7 inputs will be "clocked" into the input buffer on the positive going edge of the LBS. When in "serial load" mode (SPC = 1), serial data bits appearing at the serial input terminal, Pin 26, will be "clocked" into the input buffer on the positive going edge of HBS and LBS. (HBS and LBS must be clocked simultaneously when in "serial load" mode.)
25	HBS	High Byte Strobe. When in "parallel load" mode (SPC = 0), parallel data appearing on the DB9 (MSB) and DB8 data inputs will be "clocked" into the input buffer on the positive going edge of HBS. When in "serial load" mode (SPC = 1), serial data bits appearing at the serial input terminal, Pin 26, will be "clocked" into the input buffer on the positive going edges of HBS and LBS. (HBS and LBS must be clocked simultaneously when in "serial load" mode.)
26	SRI	Serial Input.

Pin Function Description cont'd

PIN	MNEMONIC	DESCRIPTION
27	VCC	Logic Supply. If +5V is applied, all digital inputs/outputs are TTL compatible. If +10V to +15V is applied, digital inputs/outputs are CMOS compatible.

Note 1: Logic "1" applied to a data bit steers that bit's current to the IOUT1 terminal.

PIN	MNEMONIC	DESCRIPTION
28	DGND	Digital Ground.

DAC Circuit Description

GENERAL CIRCUIT INFORMATION

The AD7522's DAC functional block consists of a highly stable Silicon Chromium thin film R-2R ladder, and ten SPDT N-channel current steering switches. Most applications require the addition of only an output operational amplifier and a voltage or current reference.

The simplified D/A circuit is shown in Figure 1. An inverted R-2R ladder structure is used – that is, the binarily weighted currents are switched between the IOUT1 and IOUT2 bus lines, thus maintaining a constant current in each ladder leg independent of the switch state.

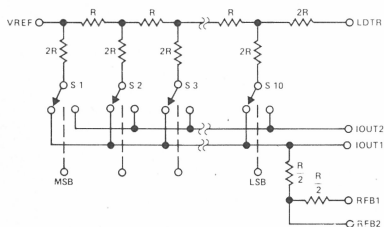



Figure 1. DAC Functional Diagram

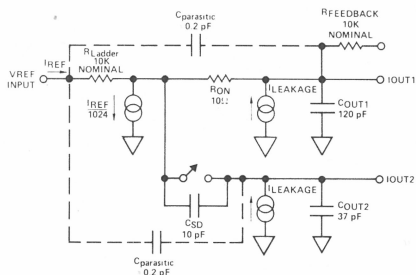



Figure 2. Equivalent Circuit (Shown for all Digital Inputs High)

EQUIVALENT CIRCUIT

The DAC equivalent circuit is shown in Figure 2. The current source $I_{LEAKAGE}$ is composed of surface and junction leakages to the substrate, while the $I_{REF}/1024$ current source represents the 1LSB of current lost through the ladder termination resistor to ground. The C_{OUT1} and C_{OUT2} output capacitances are as shown when the DAC latches feed the DAC with all "1's." If the DAC latches are loaded with all "0's," C_{OUT1} is 37 pF, while C_{OUT2} is 120 pF. In addition, C_{SD} is shunted by 10 ohms, and the 10 ohm R_{ON} in IOUT1 is replaced by a C_{SD} of 10 pF. When fast amplifiers are used, it will be necessary to provide phase compensation (in the form of feedback capacitance) to cancel the pole formed by $R_{FEEDBACK}$ and C_{OUT} if stability is to be maintained.

Applications

UNIPOLAR OPERATION

Figure 3 shows the analog circuit connections required for unipolar operation. The input code/output voltage relationship is shown in Table 1.

Zero Offset Adjustment

1. Adjust the op amp's offset potentiometer for < 1 mV on the amplifier junction. (Each millivolt of amplifier V_{OS} causes ±0.66 mV of differential nonlinearity which adds to the ladder nonlinearity.)

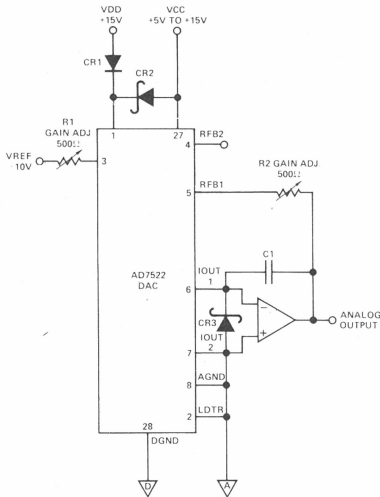



Figure 3. Unipolar Binary Operation (2-Quadrant Multiplication)

Gain Adjustment

1. Set R1 and R2 to 0Ω. Load the DAC register with all "1's."
2. If analog out is greater than −VREF, increase R1 for required full scale output. If analog out is less than −VREF, increase R2 for required full scale output.

TABLE 1
UNIPOLAR CODE TABLE

DIGITAL INPUT	ANALOG OUTPUT
1 1 1 1 1 1 1 1 1 1	$-VREF (1 - 2^{-10})$
1 0 0 0 0 0 0 0 0 1	$-VREF (1/2 + 2^{-10})$
1 0 0 0 0 0 0 0 0 0	−VREF/2
0 1 1 1 1 1 1 1 1 1	$-VREF (1/2 - 2^{-10})$
0 0 0 0 0 0 0 0 0 1	$-VREF (2^{-10})$
0 0 0 0 0 0 0 0 0 0	0

TABLE 2
BIPOLAR CODE TABLE

DIGITAL INPUT	ANALOG OUTPUT
1 1 1 1 1 1 1 1 1 1	$-VREF (1 - 2^{-9})$
1 0 0 0 0 0 0 0 0 1	$-VREF (2^{-9})$
1 0 0 0 0 0 0 0 0 0	0
0 1 1 1 1 1 1 1 1 1	$VREF (2^{-9})$
0 0 0 0 0 0 0 0 0 1	$VREF (1 - 2^{-9})$
0 0 0 0 0 0 0 0 0 0	VREF

BIPOLAR OPERATION

Figure 4 shows the analog circuit connections required for bipolar operation. The input code/output voltage relationship is shown in Table 2.

Zero Offset Adjustment

1. Adjust the offset potentiometer of amplifier A1 and A2 for < 1 mV on the respective summing junctions. If the analog out for code 1000000000 is not zero, sum current into or out of the summing junction of A1 for 0V at analog out.

Gain Adjustment

1. Load the DAC register with all "0's." Set R1 and R2 to 0Ω.
2. If analog out is greater than +VREF, increase R2 until it reads precisely +VREF. If analog out is less than +VREF, increase R1 until it reads precisely VREF.

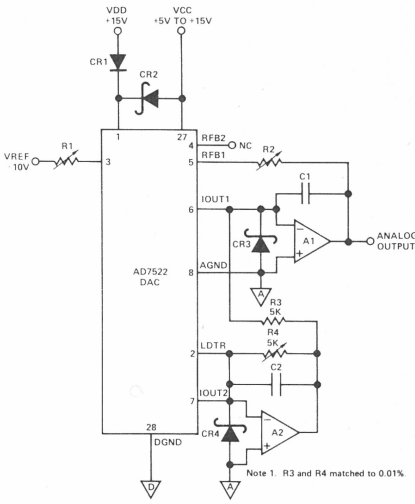



Figure 4. Bipolar Operation

SINGLE BYTE PARALLEL LOADING

Figure 5 illustrates the logic connections for loading single byte parallel data into the input buffer. DB0 should be grounded on "K" and "T" versions, and DB0 and DB1 should be grounded on "J" and "S" versions for monotonic operation of the DAC. DB9 is always the MSB, whether 8-bit, 9-bit, or 10-bit linear AD7522's are used.

When data is stable on the parallel inputs (DB0-DB9), it can be transferred into the input buffer on the positive edge of the strobe pulse.

Data is transferred from the input buffer to the DAC register when LDAC is a logic "1." LDAC is a level-actuated (versus edge-triggered) function, and must be held "high" at least 0.5μs for data transfer to occur.

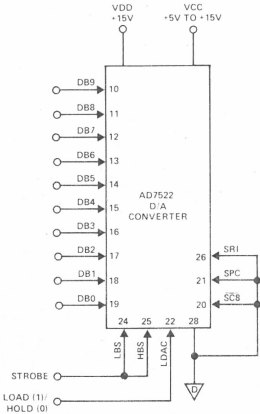



Figure 5. Single Byte Parallel Loading

TWO BYTE PARALLEL LOADING

Figures 6 and 7 show the logic connections and timing requirements for interfacing the AD7522 to an 8-bit data bus for two byte loading of a 10-bit word.

First, the least significant data byte (DB0 through DB7) is loaded into the input buffer on the positive edge of LBS. Subsequently, the data bus is used for status indication and instruction fetching by the CPU. When the most significant data byte (DB8 and DB9) is available on the bus, the input buffer is loaded on the positive edge of HBS. The DAC register updates to the new 10-bit word when LDAC is "high." LDAC may be exercised coincident with, or at any time after HBS loads the second byte of data into the input buffer.

SERIAL LOADING

Figure 8 and Figure 9 show the connections and timing diagram for serial loading.

To load a 10-bit word ($\overline{SC8}$ = 1), HBS and LBS must be strobed simultaneously with exactly 10 positive edges to clock the serial data into the input buffer. For 8-bit words ($\overline{SC8}$ = 0), only 8 positive edges are required.

The DAC register can now be loaded by holding LDAC "high."

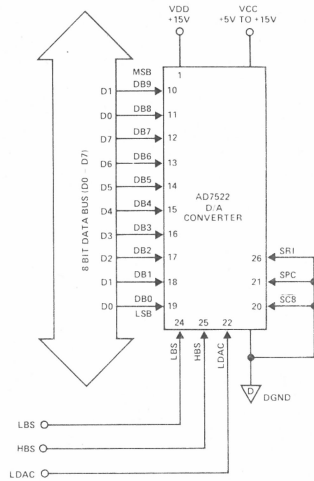



Figure 6. Two Byte Parallel Loading

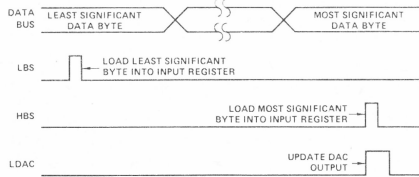



Figure 7. Timing Diagram

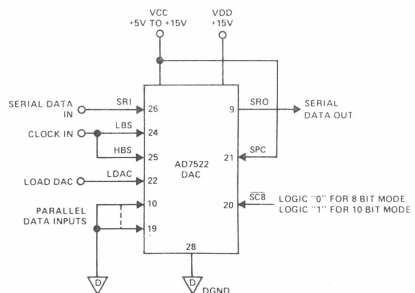



Figure 8. Serial 8- and 10-Bit Loading (Analog Outputs Not Shown for Clarity)

Applications cont'd

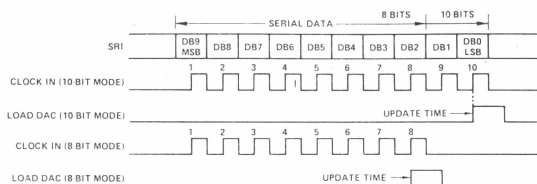
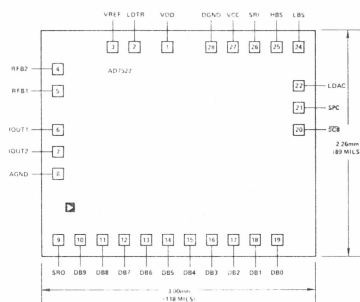
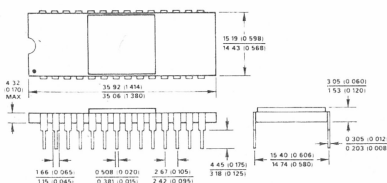
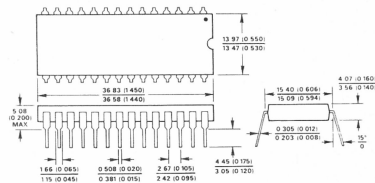



Figure 9. Timing Diagram for Serial 8- and 10-Bit Loading

APPLICATION HINTS

1. CR1 and CR2 on Figures 3 and 4 protect the AD7522 against latch-up if VCC exceeds VDD, and may be omitted if VDD and VCC are driven from the same voltage.
2. Diodes CR3 on Figure 3 and CR3 and CR4 on Figure 4 clamp the amplifier junction to −300 mV if they attempt to swing negative during power up or power down. The input structures of some high-speed op amps can supply substantial current under the transient conditions encountered during power sequencing. It is recommended that the PC layout be able to accommodate the diodes.
3. Fast op amps will require phase compensation for stability due to the pole formed by C_{OUT1} or C_{OUT2} and $R_{FEEDBACK}$.
4. During serial loading, all data inputs (DB0 through BD9), should be grounded.

Mechanical Information

BONDING DIAGRAM

28 PIN CERAMIC DIP

28 PIN PLASTIC DIP

1. Lead no. 1 identified by dot or notch.
2. Dimensions in millimeters (inches).

3½ Digit AC Line Powered DPM

AD2009

FEATURES
- AC Line Powered
- Bright, Seven Segment Gas Discharge Display
- BCD Data Outputs Standard
- Hold and Trigger Control Signals
- Full Scale Ranges of ±1.999V or ±199.9mV
- Display Blanking Control
- Industry Standard Panel Cutout

APPLICATIONS
- General Purpose DPM Applications Requiring AC Power and a High Visibility Display
- Data Logging and Digital Feedback Control Systems

GENERAL DESCRIPTION
The AD2009 is a low cost 3½ digit, AC line powered DPM designed for general purpose DPM applications. The AD2009 measures bipolar input voltages over full scale ranges of either ±1.999V or ±199.9mV, with an accuracy of ±0.1% reading ±1 digit and displays the readings on large, bright 0.55" (14mm) Beckman gas discharge displays.

LARGE, BRIGHT DISPLAY
For display only applications, the Beckman display offers excellent appearance and visibility. The AD2009 display is easily read up to 50 feet (15m) away and over all ambient lighting conditions. The non-glare lens allows a choice of either red or amber display colors, and is easily silk-screened with company logo or measurement units. External control of decimal points and display blanking is provided.

SIMPLE DATA INTERFACING
Since the AD2009 is designed around TTL logic circuits, parallel BCD data, TTL/DTL compatible, is a standard feature, allowing easy interfacing to a variety of data peripherals, such as digital comparators and line printers. Under internal control, the AD2009 converts at a nominal rate of six conversions per second. Using the Hold and Trigger controls, up to 100 conversions per second can be externally triggered.

INDUSTRY STANDARD CASE DESIGN
In response to industry's urgent need for DPM standardization, Analog Devices has adopted the most popular AC powered DPM panel cutout size for the AD2009 and all future AC line powered DPM's. Since this 3.924" x 1.682" (99.67 x 42.72mm) panel cutout is used by so many AC powered panel meters, the potential DPM customers can be assured that second-sources will be available and future new products will be usable without mechanical changes to their instruments or systems.

DESIGNED AND BUILT FOR RELIABILITY
Design and manufacturing techniques are chosen to insure reliability in the AD2009. Conservative design techniques and thorough component evaluation are only the beginning. Manufacturing processes are monitored by continuous quality assurance inspections to insure proper workmanship and testing. Like every other Analog Devices' DPM, each AD2009 is fully tested for electrical specifications, calibrated, and given one full week of failure free burn-in before shipment.

THEORY OF OPERATION
The AD2009 uses a dual slope conversion technique with an absolute value voltage to current converter input. The entire conversion cycle takes less than 10 milliseconds, allowing a complete conversion to be done during the negative half cycle of the AC line, and the resulting reading is displayed during the positive half cycle of the AC line. This scheme not only insures a flicker free display, but also allows externally triggered conversions at rates up to 100/second for data interfacing applications. In order to insure a bright display even during operation at low line voltages and to help insure the reliability of the Beckman displays, a separate power supply is provided to continually illuminate two "keep-alives" in the Beckman display.

Route 1 Industrial Park; P.O. Box 280; Norwood, Mass. 02062
Tel: 617/329-4700 TWX: 710/394-6577
West Coast 213/595-1783 Mid-West 312/894-3300 Texas 214/231-5094

SPECIFICATIONS (typical @ +25°C and nominal line voltage)

DISPLAY OUTPUT
- Beckman Seven Segment Gas Discharge Display, 0.55" High (14mm) for Three Data Digits, 100% Overrange and Negative Polarity Indication. Overload indicated by blanking the three data digits and displaying the "1" overrange. The polarity remains valid.
- Decimal Points Selectable at Input.
- Display Blanking

ANALOG INPUT
- Configuration: Bipolar, Single Ended
- Full Scale Range: ±1.999V or ±199.9mV (see S option)
- Automatic Polarity
- Input Impedance: 100MΩDC
- Bias Current, Both Ranges: 3nA @ 2V FS, 20nA @ 200mV FS
- Overvoltage Protection, Both Ranges: 200VDC Sustained

ACCURACY
- ±0.1% ±1 Digit[1]
- Resolution: 1mV or 100μV (S option)
- Temperature Range[2]: 0 to +50°C Operating
  -25°C to +85°C Storage
- Temperature Coefficient:
  Gain (both ranges) – ±60ppm/°C
  Zero Offset (2V Input) – ±30μV/°C
  (200mV Input) – ±10μV/°C
- Warm-Up Time to Rated Accuracy: 15 minutes
- Settling Time to Rated Accuracy: 0.3 sec

NORMAL MODE REJECTION
- 18dB @ 60Hz

COMMON MODE REJECTION (1kΩ source imbalance @ 50-60Hz, with standard shielded transformer)
- 2V Input – 100dB
- 200mV Input – 80dB

COMMON MODE VOLTAGE
- ±300VDC (600VAC p/p) (floated on power supply transformer when BCD outputs and control signals are not used)

CONVERSION TIME
- 10msec

CONVERSION RATE
- Internal Trigger: 6 conversions per second
- External Trigger: 0-100 conversions per second

DIGITAL CONTROL SIGNALS
- DTL/TTL Compatible

	In	Out
Logic "0"	<0.8V	<0.4V
	>2.0V	>2.4V

CONTROL INPUTS[3]
- Display Blank (1TTL Load). Logic "0" or grounding blanks the entire display, not including the decimal points. Logic "1" or open circuit for normal operation. Display blanking has no effect on output data and the display reading is valid immediately upon removal of a blanking signal.
- Hold (1TTL Load). Logic "0" or grounding disables either the external or internal trigger and the last conversion is held and displayed.
- External Trigger (1TTL Load). Positive pulse (500μsec max width) will initiate conversion.
- Decimal Points (Not TTL Compatible). Grounding will illuminate the desired decimal point. External drive circuitry must be capable of withstanding 100V when the decimal points are turned off.

DATA OUTPUTS[3]
- 3BCD Digits (Drives 6TTL Loads). Positive true, unlatched
- Overrange (Drives 6TTL Loads). Unlatched, Logic "0" indicates overrange (≥1000).
- Overload (Drives 6TTL Loads). Unlatched, Logic "0" indicates overload (≥2000).
- Polarity (Drives 6TTL Loads). Latched, Logic "1" indicates positive polarity.
- Status (Drives 10TTL Loads). All digital outputs are valid when status is at Logic "0". Logic "1" indicates conversion is in progress.
- Internal Trigger Output (Not TTL Compatible). When connected to External Trigger Input will cause the AD2009 to convert at 6 conversions per second. This output can only be used for triggering the AD2009.

POWER INPUT
- AC line, 50-60Hz, 4.2 Watts at 60Hz; 4.7 Watts at 50Hz (at nominal line voltages).

CALIBRATION ADJUSTMENTS
- Gain
- Zero
- Recommended recalibration interval – 6 months

SIZE
- 4.18"W x 1.93"H x 4.15"L (106 x 49 x 112mm)
- 4.77"L (121mm) to rear of card edge connector
- Panel cutout required: 1.682 x 3.924" (42.72 x 99.67mm)

WEIGHT
- 15 ounces (425 grams)

OPTIONS[4] – ORDERING GUIDE
- AC Power Inputs (50-60Hz)
  AD2009 – 117VAC
  AD2009/E – 220VAC
  AD2009/F – 100VAC
  AD2009/H – 240VAC
  } ±10%
-
  AD2009 – 1.999VDC Full Scale
  AD2009/S – 199.9mVDC Full Scale
-
  Lens 7 – Red with ADI Logo
  Lens 8 – Red without ADI Logo
  Lens 13 – Amber with ADI Logo
  Lens 14 – Amber without ADI Logo

CONNECTOR
- 30 Pin, 0.156" Spacing Card Edge Connector, Amphenol 225-21524-601 (117) or Equivalent
- Optional: Order AC2611 @ $4.50

PRICING
- $140 (unit quantity)
- Consult Factory for OEM quantity pricing

[1] Guaranteed @ +25°C.
[2] Guaranteed.
[3] Not to be used when the AD2009 is floating on common mode voltages.
[4] Only one input range and AC power input may be specified.
[5] Lens 7 is supplied if no lens option is specified.
Specifications subject to change without notice.

INTERFACING THE AD2009

Input Connections

The AD2009 has a single ended input with common analog and digital grounds. When digital control lines and BCD data outputs are not used, the entire DPM can be floated on the power supply transformer at up to 300VDC common mode voltages. If these signals are used, care should be taken to insure against ground loops within the system causing erratic and/or erroneous readings.

Decimal Points

Grounding the proper pin will illuminate the desired decimal point. If external logic drives are used to control the decimal points, drive circuitry must be able to withstand 100V when the decimal points are turned off.

Display Blanking

The entire display (excluding decimal points) may be blanked by applying logic "0" or grounding the proper control input (pin 13). Blanking the display has no effect on the output data or the conversion process. The data remains valid during blanking and the DPM reading is correct immediately upon removal of the blanking signal.

Interfacing Digital Data Outputs

The digital data outputs of the AD2009 are unlatched, positive true, parallel BCD, at DTL/TTL logic levels. As shown in the timing diagram (Figure 1), all data outputs are valid when the STATUS line is low. The STATUS line is high during conversion when erroneous data will be present on the outputs.

TRIGGERING CONVERSIONS

The AD2009 may be triggered internally at six conversions per second, or externally at rates of up to 100 conversions per second. For internal triggering, the Internal Trigger Output (Pin 1) should be connected to the Trigger Input (Pin B). For external triggering, a positive trigger pulse (<500μs width) should be applied to the Trigger Input (Pin B). Whether internal or external triggering is used, the last reading can be held and displayed by grounding or applying logic "0" to the Hold Input. At high conversion rates, the display may flicker unless synchronized to the AC line input, but data outputs will remain valid.

CALIBRATION PROCEDURE

"WARNING: For the safety of personnel and interconnected equipment, all calibration should be done using a plastic trimming tool only."

A precision voltage reference is needed for calibration of the AD2009. The location of calibration potentiometers is shown in Figure 2. Before calibrating the AD2009, allow the unit to warmup to normal operating temperature. Always adjust the zero offset first then the gain.

Zero adjustment: Short the signal input (Pin 2) to the signal ground (Pin 10) and adjust the zero adjustment pot until the meter reads 000.

Gain adjustment: Apply an input of +1.900V (+190.0mV on AD2009/S) and adjust the gain pot until the meter reads 1900 exactly.

Figure 1. AD2009 Timing Diagram

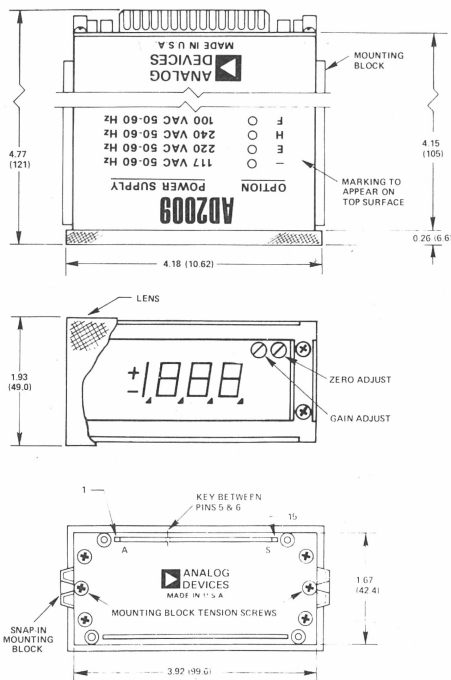



Figure 2. AD2009 Mechanical Outline (Dimensions shown in inches and (mm))

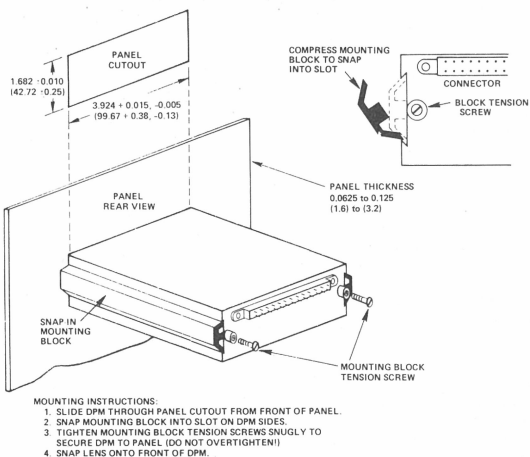



Figure 3. AD2009 Mounting Instructions
(Dimensions shown in inches and (mm))

PIN REF	PIN FUNCTION		PIN REF	PIN FUNCTION
1	INTERNAL TRIGGER OUT[1]		A	NO CONNECTION
2	SIGNAL INPUT		B	EXTERNAL TRIGGER IN[1]
3	STATUS (PRINT)		C	OVERLOAD
4	POLARITY		D	HOLD
5	BCD 8		E	BCD 1
6	BCD 2	KEY	F	BCD 4
7	BCD 80		H	BCD 10
8	BCD 20		J	BCD 40
9	BCD 800		K	BCD 100
10	SIGNAL GROUND		L	DP3/XX.X
11	BCD 400		M	DP2/X.XX
12	BCD 200		N	DIGITAL GROUND
13	DISPLAY BLANK		P	DP1/.XXX
14	OVERRANGE		R	SHIELD (EARTH GROUND)
15	AC LINE HI		S	AC LINE LO

[1] Pin 1 and Pin B must be connected for operation with internal trigger.

Figure 4. AD2009 Signal and Pin Designations

WORLD'S FIRST AUTOMATIC RANGING DIGITAL PANEL METER

MODEL DM-2000AR

Covered by GSA Contract No. GS-00S-27959

DESCRIPTION

The Datel DM-2000AR is the first digital panel meter to feature automatic ranging over three full scale input ranges. It measures readings from 20 volts down to 100 microvolts without external scaling and increases the dynamic range of the instrument to 103 db over conventional DPM's. The unique autoranging circuitry operates by first sampling an input on the most sensitive range (±199.9mV). If an overange condition is indicated, the circuitry is automatically switched to the next higher range and recycled, and again to the highest range if the second is exceeded. The worst case conversion time is 33 milliseconds (30 conversions per second) if the highest range is selected and somewhat faster on the lower ranges.

The automatic ranging feature, with three full scale input voltage ranges of ±19.99 volts, ±1.999 volts and ±199.9mV offers the user a number of advantages. The cost savings over the parts, installation and test of a front panel range switch alone would justify the use of DM-2000AR. Another advantage is the savings in operator time and the associated errors incurred with the use of a range switch. Also, the versatility of a 20 volt to 100 microvolt dynamic range cannot be discounted in bench testing situations.

The DM-2000AR has a single ended input with an input impedance of 1 Megohm and overvoltage protection to ±100 volts. Accuracy is ±.1% of full scale, ±1 count with a maximum conversion time of 33 milliseconds. The temperature coefficient of the DM-2000AR is ±100 PPM/°C.

The display of the DM-2000AR is a 3-1/2 digit, seven-segment LED display with automatic polarity indication and automatic overflow indication. A digital input allows for the testing of all readout segments by displaying "+1888". All segments are viewed through a red filter which sharpens contrast and eliminates internal reflections.

All displayed information is available at the I/O connector in BCD form. Digital inputs and outputs also allow for start-stop control and external clocking. Also, the automatic ranging can be overriden for external (manual) control. In addition, an optical isolation option is available which decouples the analog input section from the digital logic so that ground disturbances caused by the fast-switching digital circuits will not affect the analog input.

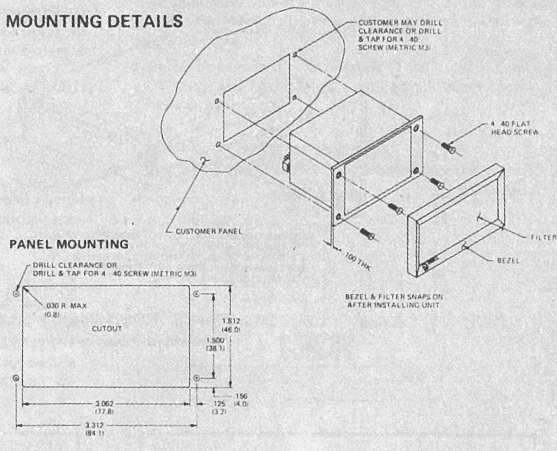
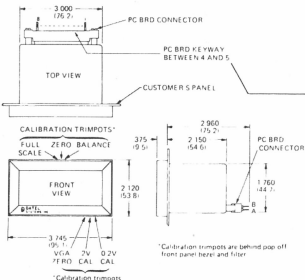
The DM-2000AR is packaged in a rugged 3"W x 1-3/4"H x 2-1/4"D LEXAN case with a total weight of less than 6 oz. Construction is entirely modular with snap-apart PC boards. The unit can easily be panel mounted with access to calibration controls obtained by snapping off the front bezel. The entire unit can be removed from its mounting panel and disassembled with just a screw driver in less than one minute.

FEATURES

- 3 Full-Auto Ranging-Scales: ±19.99V, ±1.999V, ±199.9mV
- 3-1/2 Digit Solid State LED Display
- Automatic Polarity and Overflow Display
- Operates From Single +5V DC Supply
- Optional Optically Isolated Input
- Optional External Ranging Control

MOUNTING DETAILS

PANEL MOUNTING

DRILL CLEARANCE OR DRILL & TAP FOR 4-40 SCREW (METRIC M3)
.030 R. MAX. (0.8)
CUTOUT
1.812 (46.0)
1.500 (38.1)
.156 (4.0)
.125 (3.2)
3.062 (77.8)
3.312 (84.1)

MECHANICAL DIMENSIONS INCHES (MM)

INPUT/OUTPUT CONNECTIONS

PC BRD CONNECTOR

BOTTOM	A	B	TOP	
ANALOG GROUND	1	1	ANALOG INPUT (HI)	
SHIELD GROUND	2	2	SHIELD GROUND	
NOT USED	3	3	NOT USED	
RANGE MODE	4	4	LOGIC GROUND	
$\overline{\text{DECIMAL}}$ $\overline{\text{POINT}}$ $\overline{100}$	5	5	BIT 1	OUT
$\overline{\text{DECIMAL}}$ $\overline{\text{POINT}}$ $\overline{10}$	6	6	BIT 2	OUT
$\overline{\text{DECIMAL}}$ $\overline{\text{POINT}}$ $\overline{1}$	7	7	BIT 4	OUT
$\overline{\text{E O C}}$ (STATUS)	8	8	BIT 8	OUT
$\overline{\text{OVERLOAD}}$ $\overline{\text{SCALE}}$ $\overline{\text{OUT}}$	9	9	BIT 10	OUT
INT. START GATE	10	10	BIT 20	OUT
INT. START ADJ	11	11	BIT 40	OUT
INT. START OUT	12	12	BIT 80	OUT
START INPUT	13	13	BIT 100	OUT
$\overline{\text{LAMP}}$ $\overline{\text{TEST}}$	14	14	BIT 200	OUT
1000 OUT	15	15	BIT 400	OUT
SIGN OUT	16	16	BIT 800	OUT
EXTERNAL RANGING	17	17	EXTERNAL RANGING	
POWER COMMON	18	18	POWER INPUT, +5VDC	

Edge board PC contacts are on 0.1" centers

Ground A 4 for normal autoranging operation. Jumper A12 to A13 unless using ext. clock. SEE APPLICATIONS

SPECIFICATIONS (Typical @ 25°C, 15 minutes warm-up)

ANALOG INPUT

(single ended)

Note: Display reads in volts on high ranges, millivolts on low range.

Parameter	Specification
Full Scale Input Ranges (automatic ranging)	±19.99V ±1.999V ±199.9mV
Input Bias Current	2nA (all ranges)
Input Overvoltage	±100V max.
Input Impedance	1 MEGOHM (Note 5)

Additional specifications (available only with optical isolation option):

Parameter	Specification
Input Configuration	Single ended Bipolar Floating
Common Mode Voltage	$\pm 100V_{CM}$ max. to digital output common
Common Mode Rejection	70dB @ 60 Hz

PERFORMANCE

Parameter	Specification
Accuracy	±.1% of F.S. ±1 count (5)
Resolution	±1 count (±100μvolts)
Temperature Coefficient of Full Scale	±100 PPM/°C
Zero Drift (referred to the input)	30μV/°C (199.9mV and 1.999V range) 150μV/°C (19.99V range)
Conversion Time	33 msec max (see timing diagram)
Input Settling Time	1 m sec for F.S. change
Operating Temperature Range	0°C to +50°C
Storage Temperature Range	-20°C to +85°C
Warm-up Time	15 min to specified accuracy
Input Power (See Note 4)	+5 ±.25VDC at 800mA (max), <50mV spikes

DISPLAY OUTPUT

Parameter	Specification
Display Type	red LED seven segment 0.3" high
Data	3-1/2 digits (1999 max count)
Polarity	±automatically displayed
Decimal Point	automatically displayed
Overflow	"OF" automatically displayed

DIGITAL OUTPUTS

Parameter	Specification
BCD (3LSD's) Data Outputs (pins B5 thru B16)	3 digits (8421) 12 lines positive true Loading: 2 TTL loads (2) Except 800 output: 1 TTL load
1/2 Digit (MSD) Data Output (pin A15)	1/2 digit-1 line-positive true Loading: 2 TTL loads (2)
Polarity (pin A16)	1 line, "1" = positive, "0" = negative Loading: 2 TTL loads
$\overline{\text{Overload Scale Out}}$ (pin A9) Note: Negative True	1 line, HIGH - input signal within range LOW - input signal outside range Loading: 2 TTL loads
$\overline{\text{Decimal Point Outputs}}$ Note: Negative True, Decimal points are controlled by auto-ranging logic.	3 lines, negative true Range 19.99V @ Pin 6 Range 1.999V @ Pin 5 Range 199.9mV @ Pin 7 Loading: 1 TTL load (2)
$\overline{\text{End of Conversion}}$ (pin A8, Busy)	1 line, HIGH - during the conversion period (display blanked) LOW - conversion complete Loading: 2 TTL loads (2)
Internal Start Output (pin A12)	1 line, positive true pulse Loading: 1 TTL load (2)

DIGITAL INPUTS (Dynamic inputs should have TTL rise times)

Parameter	Specification
Ranging Mode Control (pin A4)	1 line, "1" = external ranging, "0" = automatic ranging Loading: 3 TTL loads (1)
External Ranging Control (pins B17 and A17) Shown with pin A4 at HI. A17 & B17 are disabled if A4 is LO.	2 lines - Loading: 1 TTL load (1) Range / Pin A17 / Pin B17 19.99V / "0" / "1" 1.999V / "1" / "0" 199.9mV / "0" / "0" "11" Range code lights 2 D.P. Don't use.
External Start Convert Command (Pin A13) Start Trigger Clock, may be supplied from int. start, A12	1 line-min pulse width-100 nsec "0" to "1" (⎍) resets output register and blanks display, "1" to "0" (⎍) initiates conversion process. Loading: 1 TTL load (1)
Internal Start Gate (Pin A10)	1 line, gates internal clock "1" = run, "0" = stop Loading: 1 TTL load (1)
Internal Start Adjust (Pin A11)	Controls rate of Internal Start Clock (see application section)
$\overline{\text{Lamp Test}}$ (Pin A14)	1 line, negative true, displays +1888 to test all display segments Loading: sink 35mA.

NOTE: Internal start clock is 2 samples/second and is externally adjustable

ADJUSTMENTS

Zero, balance, full scale, VGA zero, 2V cal and 0.2V cal trimpots accessible behind snap-on front bezel

Parameter	Specification
Recalibration Interval	(Normal conditions) 90 days

PHYSICAL (3)

Parameter	Specification
Case Size	3"W x 1.75"H x 2.25"D
Case Material	Black polycarbonate plastic
Weight	6 oz. approx.
Mounting	Panel mounted through a 1.812" x 3.062" cutout with four 4-40 screws.
Connector (not included with DPM, add to order)	Dual 18-pin, PC Edgeboard Type, 0.1" centers (Viking #3VH18/IJN-5 or equal) with key between pins 4 and 5

PRICE (1-9)

$169. (no optoisolation, less connectors)

NOTES:

(1) Low ("0") ≤ + 0.8V, High ("1") ≥ + 2.0V — Inputs

(2) Low ("0") ≤ + 0.4V, High ("1") ≥ + 2.4V — Outputs

(3) Module is fully repairable and features snap together PC Boards

(4) Avoid logic spikes entering DPM on the +5V power input. Use external filtering if required.

Recommended power supply is a Datel UPM-5/1000B or equivalent highly regulated type. Power current is 400 to 800mA max depending on digits displayed and sample rate

(5) 1 Megohm input resistor is in series with input summing junction. Accuracy specification will degrade with significant external source impedance (> 1KΩ)

GROUND A4 FOR AUTORANGING. JUMPER A12/A13 FOR INTERNAL START CLOCK

DM-2000AR DPM TIMING DIAGRAM

AUTORANGING MODE (PIN A4 GROUNDED)
19.0 VOLTS ANALOG INPUT

NOTES:

1. TIMING IS NOT TO SCALE
2. TIMING IS TYPICAL EXCEPT WHERE INDICATED MIN OR MAX
3. EOC WILL FALL 4.5 TO 33mSec TYP AFTER START CONVERT DEPENDING ON ANALOG INPUT LEVEL
4. DUTY CYCLE OF DISPLAY BLANKING DURING EOC HIGH VARIES WITH SAMPLE RATE AND INPUT RANGE; MAY CAUSE DISPLAY FLICKER IN SOME APPLICATIONS. IF AN EXTERNAL START TRIGGER CLOCK IS USED, KEEP DUTY CYCLE LOW TO AVOID DISPLAY BLANKING AND FLICKER.

COMPLETE MODULAR CONSTRUCTION

Total modular construction is another plus for the DM-2000 AR.

Servicing is simple and straightforward.

The unit can be removed through the front panel without opening the users' instrument. Once the snap-on front bezel and four mounting screws are removed, the meter slides out of its Lexan case. The procedure is uncomplicated and takes less than one minute. Once removed, modular service is possible due to the five plug-in interconnected PC boards – no wiring or soldering is required.

Troubleshooting is accomplished through board substitution and servicing can be completed within five minutes.

A full complement of replacement boards are readily available from Datel's Service Department.

ALPHA NUMERIC INDICATION OF OVERFLOW

When the voltage input exceeds full scale by a minimum of one least significant digit, the characters "OF" are displayed. All data digits are blanked. An example of this would be when full scale is +19.99V, then +20.00V would be the smallest possible overload.

BUILT IN DISPLAY TEST FOR PERIODIC TESTING

Testing for faulty display segments can be achieved in a matter of moments guarding against erroneous readings. Grounding pin 14 at the rear connector will display +1888 to test all possible segment combinations.

OPTICALLY ISOLATED INPUT

The DM-2000AR autoranging Digital Panel Meter normally uses a single-ended input with 1 megohm impedance to the input amplifier summing junction. A common bus, (pins A1, A2, B2, B4 and A18) reference analog input ground, digital output ground and +5V power common.

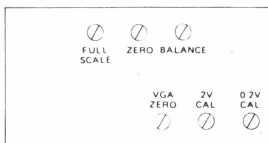
The DM-2000AR-2 version includes optoisolators and a DC/DC converter to give transformer isolation of the analog input up to ±100 Volts to power common. However, the digital outputs are still referenced to the power common/logic ground bus. Differential input impedance of the optoisolated DM-2000AR-2 is 1 megohm.

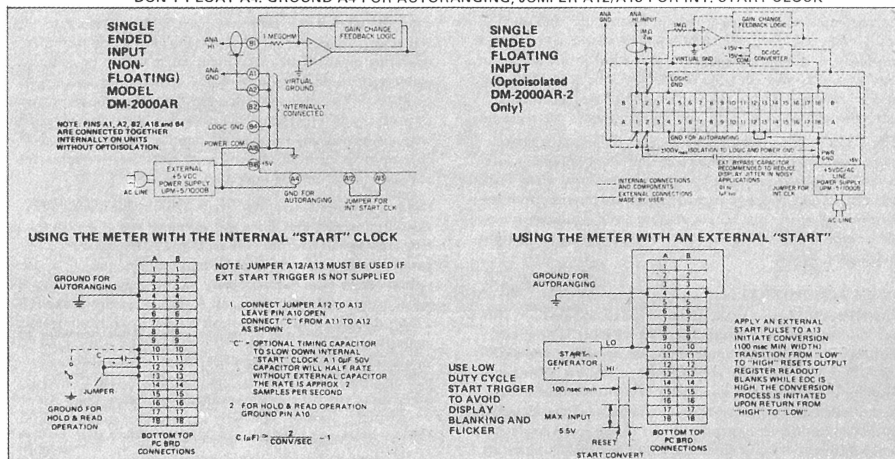
The DM-2000AR-2 isolated DPM reduces false readings in common mode voltage applications.

CALIBRATION PROCEDURE

1. Select manual mode (tie A4 to +5V) and the 20V range. Adjust the "balance" pot to obtain a flickering ± sign on the display.
2. Select the 0.2V range, and adjust the "V.G.A. zero" pot to obtain a flickering sign.
3. Repeat steps 1 and 2 until a flickering sign is obtained on both the 0.2V and 20V ranges (takes 2-3 adjustments).
4. On the 0.2V range, apply an input of ±300μV (be careful of noise on such a small input). Adjust the "zero" pot to obtain a reading of ±003 ±1 digit.
5. Select auto mode (ground A4) and apply input of +18.00V from a precision voltage reference source. Adjust "full scale" to obtain correct reading.
6. Apply input of +1.800V. Adjust "2VCAL" to obtain correct reading.
7. Apply input of +.1800V. Adjust "0.2VCAL" to obtain correct reading.

This completes calibration. Small drifts in the zero can be adjusted with the "V.G.A. zero" pot only which will not require selection of manual mode.

Location of trim pots

View behind front Bezel and Filter

DON'T FLOAT A4. GROUND A4 FOR AUTORANGING, JUMPER A12/A13 FOR INT. START CLOCK

ORDERING INFORMATION

Model DM-2000AR

PRICES (1–9 quantity)

DM-2000AR (no optoisolation, less connectors)	$169.00
DM-2000AR-2 (with optoisolation, less connectors)	$218.00

Covered by GSA Contract No. GS-00S 27959

Connectors (not included with DPM. Be sure to add to your DPM order)

Solder Tab, Datel #2335-1 (Viking 3VH18/1JN-5)	4.95
Wire Wrap, Datel #2335-2 (Viking 3VH18/1JHD-5)	4.95

Suggested AC power supply:

UPM-5/1000 B 5V, 1A, 115VAC input	$ 49.00
UPM-5/1000 BE 5V, 1A, 230VAC input	$ 54.00
Power supply Socket, MS-7	$ 3.50

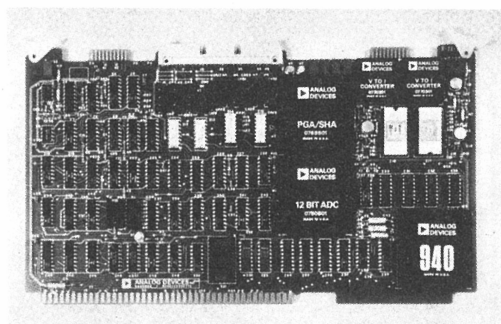
DATEL SYSTEMS, INC.
1020 TURNPIKE STREET, CANTON, MASS. 02021 / TEL. (617) 828-8000 / TWX: 710-348-0135 / TELEX: 924461
(714) 835-2751
(408) 733-2424
(213) 933-7256
(301) 840-9490

PRICES AND SPECIFICATIONS SUBJECT TO CHANGE WITHOUT NOTICE

3/77 BULLETIN MARGJ10703

Real-Time Analog I/O Interface

MODEL RTI-1200

FEATURES
- Complete Analog I/O Subsystem
- Intel SBC-80/10, 80/20, and MDS Compatible
- Memory Mapped I/O Interface
- Data Acquisition:
  - Up to 32 Input Channels
  - Sample and Hold Amplifier
  - Programmable Gain Amplifier
  - 12 Bit A/D Converter
  - Input Fault Protection
- Real-Time Pacer Clock System
- On-Board PROM Socket
- Two Optional 12 Bit DAC's
- Optional 4-20mA Current Outputs
- Optional Single +5V Power
- Memory Overlay -- RAM and ROM Inhibit

GENERAL DESCRIPTION

The RTI-1200 is a complete analog input/output subsystem that greatly simplifies the task of interfacing analog signals to Intel SBC-80 Single Board Computers, or other 8080-based microcomputers. It is functionally, electrically, and mechanically compatible with the SBC-80, and all connections to it are made simply by plugging the RTI-1200 into a slot in a card cage that also contains an SBC-80. The RTI-1200 can also be readily interfaced to other 8080-based microcomputers whose address, data, and control busses are accessible.

The RTI-1200 is interfaced to an SBC-80 or other 8080 based microcomputer as a block of contiguous memory locations. It combines on a single printed circuit card many features and capabilities which reduce the hardware required to interface analog signals to a microcomputer, and significantly ease the programming effort associated with inputting and outputting analog signals.

DATA ACQUISITION

The RTI-1200's most basic function is data acquisition. This is accomplished with an analog input multiplexer, a programmable gain amplifier, a sample-and-hold amplifier, and a 12 bit A/D converter. These components are shown in the block diagram (Figure 1). The standard RTI-1200 offers either 16 single ended or 8 differential input channels (user selected). An optional multiplexer expander allows for up to 32 single ended or 16 differential input channels. All of the analog inputs are fully protected up to ±28 volts, and additional protection against larger, potentially destructive overloads is afforded by fusing resistors located at the inputs.

The RTI-1200's A/D Converter can be configured by the user to accept 0 to +10V, ±5V, or ±10V full scale input signals. A programmable gain amplifier preceding the A/D converter has software selectable gains of 1, 2, 4 and 8. This expands the dynamic range of the A/D converter to 15 bits, and results in greater input sensitivity. For example, when operating on the 0 to +10V input range with a programmable gain amplifier gain of 8, the actual input range is 0 to +1.25V. The programmable gain amplifier allows the user to program different gains for different input channels, or to have different gains for varying input levels on the same channel. It is even possible to write software to implement automatic gain ranging operation.

Eight of the input channels have provisions for resistors provided by the user that allow the inputs to accept 4–20mA current loop signals. Output data from the A/D converter is in natural binary code for unipolar input ranges, and at the user's option can be either offset binary or two's complement coding when using bipolar input ranges. A special feature of the RTI-1200's data acquisition operation is that the controlling microcomputer's CPU (i.e., the 8080) is not tied up while a conversion is taking place. This significantly enhances system throughput capability and flexibility, as the CPU is free to pursue other tasks while an A/D conversion is in progress.

Route 1 Industrial Park; P.O. Box 280; Norwood, Mass. 02062
Tel: 617/329-4700 TWX: 710/394-6577
West Coast 213/595-1783
Mid-West 312/894-3300
Texas 214/231-5094

SPECIFICATIONS (typical @ +25°C and with +5V and ±15V, unless otherwise noted)

DATA ACQUISITION	
Number of Analog Inputs	
Standard	16 Single-Ended or 8 Diff.
With Multiplexer Expander [1]	32 Single-Ended or 16 Diff.
Multiplexer Switching Characteristics	Break-Before-Make. All Switches Open When Power is Off.
Input Voltage Ranges[2]	0 to +10V, ±5V, ±10V
Programmable Gains[3]	1, 2, 4, 8 Software Selectable
Input Impedance	$>10^9$ Ohms
Input Bias Current	
at +25°C	5nA
over 0 to +70°C	50nA
Diff. Input Bias Current	
at +25°C	3nA
over 0 to +70°C	3.5nA
Input Overvoltage Protection	
Continuous Overvoltage	±28 Volts maximum
Overvoltage >±28V	Fusing Resistors
Accuracy	
Resolution	12 Bits
Nonlinearity Error[4]	±1/2LSB typ, ±1LSB max
Diff. Nonlinearity Error	±1/2LSB typ, ±1LSB max
Quantization Error	±1/2LSB max
Input Offset Voltage[5]	Adjustable to Zero
Gain Error[5]	Adjustable to Zero
CMRR	75dB min
CMV	±10V
Noise Error[6]	±1/2LSB max
Temperature Coefficients	
Gain	±15ppm/°C typ, ±25ppm/°C max
Offset	±25μV/°C Referred to Input
Diff. Nonlinearity	±3ppm/°C max
Settling Time to ±0.01%[7]	10μs max at any Gain
SHA Aperture Time	90ns
SHA Aperture Width	20ns
SHA Aperture Uncertainty	±5ns
Conversion Time	25μs max
Maximum Throughput Rate[8]	28kHz
ANALOG OUTPUTS	
Number of DAC Channels[9]	2
Accuracy	
Resolution	12 Bits
Nonlinearity Error[4]	±1/2LSB
Diff. Nonlinearity Error	±1/2LSB
Voltage Output Characteristics	
Voltage Output Ranges[2]	±2.5V, 0 to +5V, ±5V, 0 to +10V, ±10V
Output Current	5mA min @ ±10V
Settling Time [10]	10μs max
Gain TC	±8ppm/°C typ, ±15ppm/°C max
Offset TC	±5μV/°C typ, ±20μV/°C max
Current Loop Characteristics[11]	
Current Output Range	4 to 20mA
Load Resistance Range	0 to 500Ω
Loop Supply Voltage	+15V to +30V
Settling Time [12]	50μs max
Gain TC	±10ppm/°C typ, ±25ppm/°C max
Offset TC	±0.4μA/°C
Reference Voltage Output	+5.00V ±0.02% @ 5mA max
REAL-TIME PACER CLOCK SYSTEM	
Modes of Operation	Pacer-Timed Conversion Trigger, Pacer-Timed Interrupt, Pacer Off
Types of Clocks	Variable Frequency R-C, Fixed Frequency Crystal, External
Crystal Clock Freq. [13]	Determined by User Supplied Crystal
Variable Freq. Clock Range	30Hz to 30kHz, User Adjustable
LOGIC DRIVER OUTPUTS	
Number Available	2
Characteristics	Open Collector, 30V max, 300mA max Continuous Sink Current per Output
MICROCOMPUTER INTERFACE	
Compatibility	Completely Compatible with Intel SBC-80/10, SBC-80/20, and MDS Bus System
Type of Interface	Interfaces as a Block of Memory Locations, Using Address, Data and Control Busses.
Position in Memory[14]	User Selectable Among any of 14 Possible Locations.
On-Board PROM	Socket for Intel 2708 or Equivalent 1024 Byte x 8 Bit PROM, of which 1008 Bytes are Usable.
Memory Overlay (Inhibit)	RAM or ROM
POWER REQUIREMENTS[15, 17]	
Without DC/DC Option [16]	+15V ±3% @ 40mA -15V ±3% @ 40mA +5V ±5% @ 1.2A
With DC/DC Option	+5V ±5% @ 1.7A
TEMPERATURE RANGE	
Operating	0 to +70°C
Storage	-55°C to +85°C
MECHANICAL	6.75" x 12.00" with 0.6" Board-to-Board Spacing (171.5 x 304.8 x 15.24mm)

NOTES

[1] The multiplexer expander is an option, and is shown in the ordering guide as MUX EXP.

[2] The desired range is user selectable with straps.

[3] The input gain of a channel is multiplied by the gain setting of the programmable gain amplifier (e.g., the input range of the 0 to +10V range when using a gain of 8 is 0 to +1.25V).

[4] Defined as deviation from a straight line passing through the end points of the range.

[5] For any one software programmable gain setting. Maximum offset shift of ±1LSB or gain shift of ±0.02% when using a programmable gain setting other than the one used during calibration.

[6] When using a programmable gain setting of 1. It is ±1.5LSB max when using a programmable gain setting of 8.

[7] For a 20V step. This specification is valid for a step change on one input, or following a channel change, or following a programmable gain change, or simultaneous changes involving any combination of these changes.

[8] Based on a 10μs settling time, followed by a 25μs A/D conversion time. Overall system throughput rate is enhanced because the CPU is not held up during conversions.

[9] Two channels of D/A converters and two channels of current loops are available on an option basis. See Ordering Guide.

[10] To ±0.01% of full scale range following a 20V step.

[11] The current loop characteristics include the effects of the driving D/A converter.

[12] To ±0.02% of full scale current following a full scale step.

[13] Space provided for HC-18/U crystal cut for a frequency of up to 50MHz. User can select to divide crystal frequency by 10^3 or 10^4 on-board the RTI-1200.

[14] The memory map on page 4 shows in detail where the data and control functions appear in memory.

[15] Power requirements shown are for an RTI-1200 with no DAC or current loop options.

[16] The DC/DC power converter is an option that converts +5VDC power to ±15V. It is shown in the ordering guide as DC/DC.

[17] +12V and -5V power is required only if optional PROM is used. This power is supplied by the SBC bus.

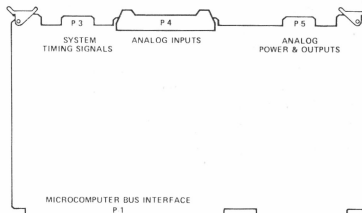
Specificatons subject to change without notice.

RTI-1200 ORDERING GUIDE

MODEL NUMBER	OPTIONS: DC/DC	MUX EXP	DACS	CURRENT LOOPS	PRICE (1–9)
RTI-1200-001					$629
-004			X		779
-006			X	X	877
-011		X			664
-014		X	X		814
-016		X	X	X	912
-101	X				696
-104	X		X		846
-106	X		X	X	944
-111	X	X			731
-114	X	X	X		881
-116	X	X	X	X	979

X DENOTES OPTIONS INCLUDED WITH THE CORRESPONDING MODEL NUMBER. THE OPTIONS ARE DESCRIBED IN NOTES 1, 9, AND 16.

RTI-1200 MECHANICAL OUTLINE

P3
SYSTEM TIMING SIGNALS
P4
ANALOG INPUTS
P5
ANALOG POWER & OUTPUTS
MICROCOMPUTER BUS INTERFACE
P1

MATING CONNECTORS FOR RTI-1200

MATING CONNECTORS FOR THE RTI-1200

PART NO.	MATES TO	DESCRIPTION	PRICE (1–9)
AC1551	P3 or P5	Flat Cable Connector 20 Pin, 0.1" Center	$5.00
AC1552	P4	Flat Cable Connector 50 Pin, 0.1" Center	7.50
AC1553	P4	Flat Cable Connector 50 Pin, 0.1" Center with 2' Color Coded Assembly Attached	12.50

Figure 1. RTI-1200 Functional Block Diagram

ANALOG OUTPUTS

The RTI-1200 has provisions for two optional 12 bit D/A converters which are software driven via double buffered registers. They can be used for such functions as driving an analog recorder, or generating analog control signals. Both D/A converters can be user set to any of five voltage output ranges. The D/A input data is natural binary for unipolar output ranges, and at the user's choice can be offset binary or two's complement for bipolar output ranges. Both analog output channels can also be optionally equipped with 4-20mA current loop outputs. This permits them to drive directly the 4-20mA control loops often used in process and industrial controls.

ON-BOARD MEMORY

The RTI-1200 contains a socket which can accommodate a 1024 byte x 8 bit PROM, such as the Intel 2708. The user can store programs in such a PROM that would establish setpoints, perform data linearization, execute testing subroutines, or perform other RTI-1200 related tasks. This can be of significance in easing programming effort, particularly when more than one RTI-1200 is used with a single SBC-80. Alternatively, the PROM socket can act simply as an extension of the PROM space available on the microcomputer.

REAL TIME PACER CLOCK SYSTEM

Most real world microcomputer applications requiring interfacing to analog data also require that many operations be referenced to real time. The RTI-1200 is equipped with a highly versatile real time pacer clock system that can provide real time operation without resorting to cumbersome and grossly inefficient software timing loops. Two pacer clocks are provided. One clock is of the R-C variety, and can be set by the user to any frequency between approximately 30Hz and 30kHz. The other is a crystal controlled clock, in which a user supplied crystal generates very precisely timed pulses. These pulses can be used to generate accurately spaced A/D conversions, as is required in Fourier transform analysis, or in generating a highly accurate time-of-day clock. A pulse from either of the clocks, or an externally supplied pacer signal, can either trigger A/D conversions directly, or signal interrupts to the controlling microcomputer. The Time Mark bit in the status word is set each time a pacer tick occurs.

OTHER FEATURES

Two software driven open-collector logic driver outputs are available for system control functions, such as providing pen lift commands in an analog data recording application. In addition, a precision 5 volt reference is a standard feature of every RTI-1200 for use in calibration and testing (e.g., as a test input on one of the anlaog input channels). Finally, the RTI-1200 can be ordered with an optional DC/DC power converter. In those instances where +15V and -15V power is not readily available, this option allows the RTI-1200 to be operated solely from the same +5V supply that powers the microcomputer used with the RTI-1200.

MEMORY MAP INTERFACE

The RTI-1200 interfaces to the SBC-80 as a 1K block (1024 bytes) of memory. The SBC-80 can address 65,536 bytes of memory, which can be envisioned as 64 blocks of 1024 bytes each. The RTI-1200 can be configured by the user to occupy one of 14 selected blocks. The 14 possible blocks are spread throughout the 65K address space, so the user should have no trouble positioning the RTI-1200 in a block that does not interfere with already committed address space.

The top 16 addresses (i.e., highest numbered) in the 1K block occupied by the RTI-1200 are devoted to the data and control functions of the RTI-1200. The bottom 1008 addresses are reserved for the on-board PROM. This structure can be seen by referring to Figure 2, the memory map. The byte addresses are shown in hexadecimal notation, with the most significant digit on the left. The addresses from XFFO to XFFF are those associated with the RTI-1200 operation itself (the actual value of hexadecimal digit X is determined by where in the 8080's memory address space the user wishes to have the RTI-1200 appear. All of the memory bytes associated with a particular RTI-1200 will have the same value for X.) The bottom 1008 memory addresses, from XCOO to XFEF, are reserved for the use of an on-board PROM.

Since the RTI-1200 interfaces as memory, any of the 8080's memory reference instructions can be used. The memory map has been carefully thought out so as to make programming as easy as possible. A complete discussion of the memory map is included in the RTI-1200's User's Guide, and only a summary is included here.

When acquiring data, the desired channel is written into address XFFA. The number of the selected channel can also be read back, allowing the use of an increment memory instruction to advance the input multiplexer to the next channel. The desired gain of the programmable gain amplifier is written into address XFF9. A conversion is commanded either by a pulse from one of the pacer clocks, or by writing a convert command into address XFFB. The end of a conversion can be determined either by checking the EOC bit in the status word, or by directing the A/D converter's EOC signal to trigger an interrupt. The A/D converter's output data can be read as 12 bit data in two byte format at addresses XFFD and XFFE. This can be performed with a single LHLD instruction. If only 8 bit data is required, the 8 most significant bits can be read as a single byte at address XFF8.

Analog output data is loaded into the two 12 bit D/A converters at addresses XFF4 through XFF7. The data is in two byte form, and the data for a single D/A converter can be loaded with a single SHLD instruction. The use of double buffers on the RTI-1200 permits the two data bytes to be loaded simultaneously into the D/A converter. This allows the D/A converter's analog output to move directly from an old analog value to a new analog value without first going to an intermediate value.

Address XFFF is the address of the byte used to select one RTI-1200 from among two or more when multiple RTI-1200's time share the same 1k block of memory. Address XFF3 contains the two bits that control the two logic driver outputs. The exact nature and use of the status and setup bytes, as well as hints on maximizing program efficiency when using the RTI-1200, are covered in the RTI-1200 User's Guide.

BYTE ADDR	7	6	5	4	3	2	1	0	BYTE NAME	OPERATION
XFFF	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	S_3	S_2	S_1	S_0	CARD SELECT	WRITE
XFFE	0 / $\overline{MSB}$	0 / $\overline{MSB}$	0 / $\overline{MSB}$	0 / $\overline{MSB}$	MSB / $\overline{MSB}$	B_{10}	B_9	B_8	ADC DATA	READ
XFFD	B_7	B_6	B_5	B_4	B_3	B_2	B_1	LSB	ADC DATA	READ
XFFC	END OF CONV	BUSY	CONV. DLYD	MSB	B_{10}	$\emptyset$	$\emptyset$	TIME MARK	STATUS	READ
XFFB	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	CONVERT COMMAND	WRITE
XFFA	$\emptyset_0$	$\emptyset_0$	$\emptyset_0$	A_4	A_3	A_2	A_1	A_0	MUX ADDR	WRITE, READ
XFF9	$\emptyset_0$	$\emptyset_0$	$\emptyset_0$	$\emptyset_0$	$\emptyset_0$	$\emptyset_0$	G_1	G_0	GAIN SELECT	WRITE READ
XFF8	MSB / $\overline{MSB}$	B_{10}	B_9	B_8	B_7	B_6	B_5	B_4	ADC DATA	READ
XFF7	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	MSB / $\overline{MSB}$	B_{10}	B_9	B_8	DAC 1 DATA	WRITE
XFF6	B_7	B_6	B_5	B_4	B_3	B_2	B_1	LSB	DAC 1 DATA	WRITE
XFF5	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	MSB / $\overline{MSB}$	B_{10}	B_9	B_8	DAC 2 DATA	WRITE
XFF4	B_7	B_6	B_5	B_4	B_3	B_2	B_1	LSB	DAC 2 DATA	WRITE
XFF3	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	D_1	D_0	DRIVERS	WRITE
XFF2	NOT USED								—	—
XFF1	NOT USED								—	—
XFF0	EOC INT EN	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	PACER FCN	PACER SELECT		SETUP	WRITE
XFEF TO XCOO	ON BOARD PROM WITH 1008 USABLE BYTES									

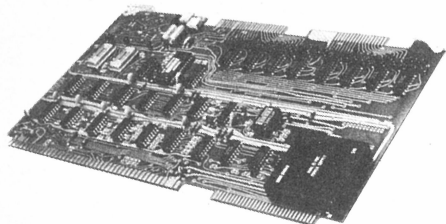
NOTES:
1. THE SYMBOL $\emptyset$ MEANS THE BIT IS IGNORED DURING A WRITE OR IS INDETERMINATE DURING A READ.
2. THE SYMBOL $\emptyset_0$ MEANS THAT THE BIT IS IGNORED DURING A WRITE, AND READ AS A 0 DURING A READ.
3. BITS SHOWN AS ⊟ HAVE THE UPPER VALUE FOR UNIPOLAR CODES, AND THE LOWER VALUE FOR BI-POLAR CODING.

Figure 2. RTI-1200 Memory Map

CARD SELECT FEATURE

The RTI-1200 contains a card select feature, which if enabled by the user, allows up to 15 RTI-1200's to share a single 1k block of memory locations in the SBC-80. This feature (which is somewhat analogous to memory paging, or memory bank selection) allows one RTI-1200 to be active while the others are in a standby or wait state. This feature can be very useful in simplifying software when it is desirable to use the same subroutines with more than one RTI-1200 in a given system. It also conserves the use of memory space.

RTI-1200 USER'S GUIDE

Detailed installation and operating information, along with programming hints and a technical explanation of the operation of the RTI-1200 are contained within the RTI-1200 User's Guide. A copy of this manual is shipped with each RTI-1200. It is also available separately from Analog Devices for $5.00.

microcomputer peripherals/power supplies
signal conditioning components/amplifiers
data acquisition systems/data converters

MP8600 SERIES

BURR-BROWN
ANALOG I/O SYSTEMS
FOR INTEL MICROCOMPUTERS

- INTEL - SBC80 Compatible
- INTEL - Intellec MDS Compatible

- LOW COST - $198 (100's)
- EASY TO PROGRAM
  Systems are treated as memory
- REDUCES SYSTEM DEVELOPMENT TIME
  System engineered and specified
  Operates from Computer's +5VDC
  Power Supply if desired
- EASY TO USE
  8 to 64 input channels on one board
  Analog input and output on one board

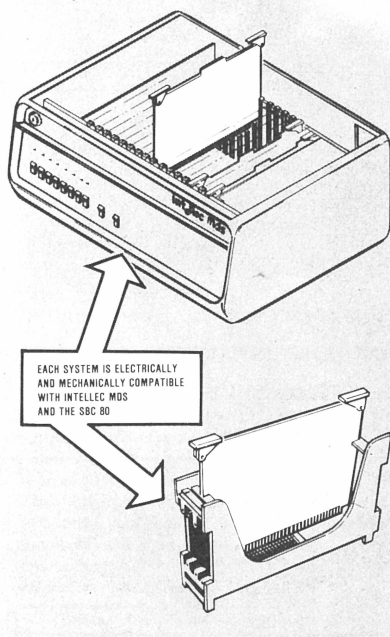
BURR-BROWN

BB

International Airport Industrial Park • P.O. Box 11400 • Tucson, Arizona 85734
Tel: 602-294-1431 • Twx: 910-952-1111 • Cable: BBRCORP • Telex: 66-6491

DESCRIPTION

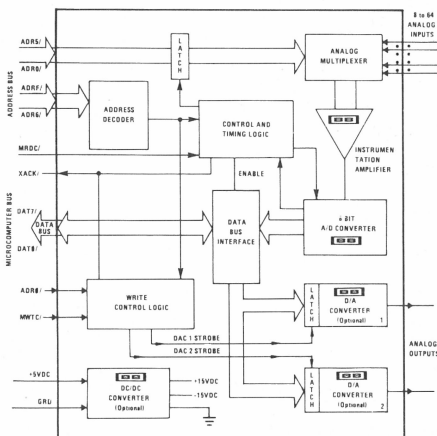
These microcomputer peripherals provide two functions that interface directly to Intel's SBC80 and Intellec MDS microcomputers. The functions are: (1) Analog Data Acquisition and (2) Analog Output. The devices are electrically and mechanically compatible with any SBC80 and Intellec MDS. Both analog input and output systems are contained on a single printed circuit board that is treated as memory input or output by the CPU. The cards will mate to any memory or I/O slot. They are compatible with the 0.6" spacing of the SBC80 or the 0.75" spacing of the Intellec MDS. The analog interface for each system is a connector at the opposite edge of the board from the bus connector.

The Data Acquisition system is available with up to 64 channels single-ended on one board. It includes an input multiplexer, high gain instrumentation amplifier, 8-bit A/D converter along with all the necessary timing, decoding and control logic. A DC/DC converter (+5V to ±15V) is also available so that only the computer's power supply is required. The Data Acquisition System is available with two optional 8-bit D/A converters to provide analog input and output on the same board.

THEORY OF OPERATION

When programming with these peripherals, they are treated as memory locations. Any memory reference instruction can be used. Both the A/D converter output and the D/A converter input are 8-bit words so one memory location is needed for each channel. Because the address block occupied by each peripheral is user selectable, it can be placed anywhere in memory.

Because these units are treated as memory, a minimum of instructions are needed to read an input channel or to set the input of a D/A converter. For instance, the LHLD (load) instruction followed by the proper address can be used to read data from two successive analog input channels. It will automatically select the desired channel, initiate conversion and when conversion is complete, transfer the A/D converter output for the first channel to the 8080's L register and the second channel to the H register. Likewise a single LDA instruction can be used to read one analog input channel.

All of these systems are jumpered at the factory with the first channel at address $F700_{16}$. Each subsequent channel is one memory location past the start of the last channel so that the second channel is at location $F701_{16}$.

ANALOG INPUT/ OUTPUT SYSTEM

SPECIFICATIONS

(All specifications typical at 25°C unless otherwise noted.)
Prices and specifications subject to change without notice.

ANALOG INPUT/OUTPUT SYSTEM

ANALOG INPUT	
Number of analog inputs	
8 differential	MP8608
16 single-ended	MP8616
32 differential or 64 single-ended [5]	MP8632
Input voltage range[1]	±10mV to ±5V
ADC gain ranges[1] (strap selectable)	±10V, 0 to 10V, 0 to 5V ±5V, ±2.5V
Amplifier gain range[1] (resistor programmable)	1 to 1000 V/V
Amplifier gain equation	$G = 100k\Omega / R_{EXT}$ (Resistor programmable)
Input overvoltage protection	±15V
Input impedance	100 megohms
Bias current	
25°C (max)	+300nA
0°C to 70°C	-2nA/°C
Amplifier input offset voltage	±2mV
Amplifier input offset voltage drift	$\pm(5 + \frac{1000}{G})$ μV/°C

ANALOG INPUT TRANSFER CHARACTERISTICS	
Resolution	8 bit binary
Throughput accuracy ±5V range (max)	±0.4% FSR[2]
±10mV range	±0.5% FSR
Temperature coefficient of accuracy	
±5V range (max)	±0.02% FSR/°C
±10mV range	±0.07% FSR/°C
Conversion time ±5V range	44 microseconds
±10mV range	84 microseconds
CMRR (for differential inputs)[3]	66 dB (Gain = 2)
	86 dB (Gain = 100)

ANALOG OUTPUT	
Number of analog outputs	2
Output voltage range[4]	±10V, 0 to 10V, ±5V, 0 to 5V, ±2.5V at 5mA (strap selectable)
Output impedance	1Ω
Output settling time (max)	< 5 microseconds

ANALOG OUTPUT TRANSFER CHARACTERISTICS	
Resolution	8 bits binary
Throughput accuracy (max)	±0.4% FSR
Temperature coefficient of accuracy	
Unipolar	±0.005% FSR/°C
Bipolar	±0.01% FSR/°C

DIGITAL INPUT/OUTPUT	
All signals are compatible with Microcomputer Bus	
Output coding	Bipolar, two's complement; Unipolar, straight binary
An analog input channel is selected by:	ADR0 through ADR5
An analog output channel is selected by:	ADR0
The input/output data bits are read through:	DAT0 through DAT7

POWER REQUIREMENTS	
MP8608, MP8616, MP8632,	+5VDC +5% at 1 amp, 25mV ripple
MP8608-NS, MP8616-NS, MP8632-NS	+5VDC ±5% at 500mA, 25mV ripple +15VDC ±3% at 40mA, 5mV ripple -15VDC ±3% at 40mA, 5mV ripple
With analog output	
MP8608-AO, MP8616-AO, MP8632-AO	+5VDC ±5% at 2 amp, 25mV ripple
MP8608-AO-NS, MP8616-AO-NS, MP8632-AO-NS,	+5VDC ±5% at 500mA, 25mV ripple +15VDC ±3% at 100mA, 5mV ripple -15VDC ±3% at 100mA, 5mV ripple

TEMPERATURE RANGE	
	0°C to 70°C

OPERATING INSTRUCTIONS

INSTALLATION

These units are shipped from the factory calibrated and ready for immediate use. Installation requires only plugging the card into any empty slot in the computer and wiring the analog connector.

PROGRAMMING

Programming of this analog I/O board is easily accomplished since all channels are treated as memory locations. Any memory reference instruction can be used. A single STA instruction may be used to load the accumulator contents to one of the D/A converters. Likewise a single LDA instruction can be used to read an analog input channel.

Single instructions can also be used to set the inputs of both D/A converters and read two adjacent analog input channels. An SHLD instruction referenced to DAC 1 will load the contents of the L register into DAC 1 and the contents of the H register into DAC 2. An LHLD instruction will read the channel addressed and the next higher channel. The channel addressed will be transferred to the L register and the next higher channel to the H register. Of course, any MOV instruction may also be used if direct addressing is not desired.

The normal operation of this board halts the CPU during the conversion time of the analog input system. This is because the software in this mode is simpler than in any other (i.e., only one instruction required!). If the halt feature is not desirable, it may be disabled. Figure 1 shows the jumpers required. The jumpers shown with an asterisk are plated-through holes and must be drilled out before installation of the other jumpers. A 0.055″ (No. 54) drill should be used for this purpose. Caution must be exercised to prevent damage to the board (see Figure 2).

MECHANICAL CHARACTERISTICS

Compatible with Intellec MDS and SBC-604/614 card spacing.

Minimum card spacing: 12.7mm (0.5″).

Microcomputer bus connector required: 86 pin PC edge connector with 0.156″ contact centers.

50 pin analog edge connector on board.

Mating connector available from Burr-Brown: 2350MC (Viking # 3VH25/1JN5, solder tab); from 3M: 3415-0001 (Scotchflex).

(1) Connected at the factory for ±5V range (ADC range = ±10V, Gain = 2).
(2) FSR is Full Scale Range (i.e., 10V for ±5V range).
(3) DC to 60 Hz with 1 kΩ source unbalance.
(4) Connected at the factory for ±10V range.
(5) Connected at the factory as 32 differential.

OPERATION WITH HALT. JUMPERS REQUIRED	OPERATION WITHOUT HALT. JUMPERS REQUIRED
W55*, W56*, W57*	W53, W54, W58

FIGURE 1. Halt Selection Jumpers

For operation without halting the CPU, the conversion should be started by using a single channel memory reference instruction (LDA or MOV). Then the CPU should execute a routine which will take longer than the conversion time (44 to 84 microseconds). When the CPU now uses an LDA or MOV referenced to the same memory location, the converted data will be transferred to the CPU.

The voltage data for these boards is represented by an 8-bit two's complement binary number. With a ±5V range, each bit has a value of 39.1mV, with the polarity of the voltage indicated by the sign of the binary number.

Each board is set at the factory for a block of addresses beginning at F700. Any analog data channel requires one memory location. Thus the first analog channel is located at F700 while the second analog channel is located at F701.

ADDRESS MODIFICATION

The base address of a board can be set to any value by properly jumpering its address selector. The most significant 8 bits of the address (ADR/8-F) are jumpered to read F7 by plated through connections on all boards. These addresses can be changed by first drilling out the hole that makes the connection (Figure 2) and then soldering a wire jumper between the bit and logical zero or one. A 0.055" (No. 54) drill should be used for this purpose. Caution must be exercised to prevent damage to the board and the scattering of metal particles over its surface.

The remaining lower ordered bits have been connected by wire jumpers at the factory. To change the sense of a bit simply reverse the connection of its jumper.

ADDRESS	HIGH	LOW
4	W37	W38
5	W35	W36
6	W33	W34
7	W31	W32
8	W29*	W30
9	W27*	W28
A	W25*	W26
B	W23	W24*
C	W21*	W22
D	W19*	W20
E	W17*	W18
F	W15*	W16

* Plated through jumpers

ADDRESS JUMPERS

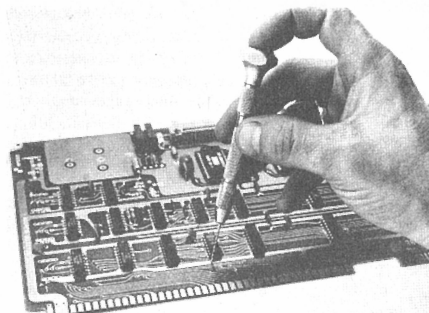



FIGURE 2. Drilling Out Plated Through Holes

ANALOG OUTPUT RANGE SELECTION

When included, each DAC is jumpered at the factory for ±10 volt operation (two's complement coding). However, it is possible to alter these jumpers as shown in Figure 3 for other output voltages and coding. Jumpers indicated by an asterisk are plated through holes on the board and should be removed by drilling as described in the section on address modification. When making a change, first remove those jumpers indicated for the present range and replace them with those jumpers required for the desired range.

Range	DAC 1	DAC 2
±10	W66*, W67*	W70*, W72*
±5	W65, W67*	W69, W72*
±2.5	W65, W67*, W74	W69, W72*, W73
0 to +10	W65	W69
0 to +5	W65, W74	W69, W73
Coding		
Two's Complement	W61*	W63*
Straight Binary	W62	W64

FIGURE 3. Analog Output Range Selection

Two's complement coding is typically used for bipolar ranges and straight binary for unipolar ranges, but either coding can be used for any range.

BIPOLAR - TWO'S COMPLEMENT			
Digital Input Output	±10V	±5V	±2.5V
01111111 (7F$_{16}$)	+9.922V	+4.961V	+2.480V
10000000 (80$_{16}$)	-10.000V	-5.000V	-2.500V

UNIPOLAR - STRAIGHT BINARY		
Digital	Input Output	0 to +10V
11111111 (FF$_{16}$)	9.961V	4.980V
00000000 (00$_{16}$)	0.000V	0.000V

TABLE I. Analog Full Scale Range Values.

ANALOG INPUT RANGE SELECTION

The analog input system can be set for any range between ±5V and ±2.5mV. It is set for ±5V (two's complement

coding) from the factory. There are two gain determining elements in this system: the A/D converter and the instrumentation amplifier (IA). The A/D converter is set for a ±10V range and the IA for a gain of 2 at the factory. The A/D converter can be set for other ranges simply by changing jumpers as shown in Figure 4. Before adding new jumpers, remove those indicated for the present range. The input voltage presented to the analog multiplexer must not exceed 5.25VDC for proper operation.

RANGE	JUMPERS
±10V	W1*, W2*
±5V	W2*, W4
±2.5V	W2*, W4, W5
0 to +10V	W3, W4
0 to +5V	W3, W4, W5

*Plated through jumpers

FIGURE 4. A/D Converter Range Setting Jumpers

As configured at the factory, this board is jumpered for two's complement operation (see Table I above) with jumper W59* inserted and W60 open. For operation in the straight binary mode (any range) jumper W59* is open and W60 is inserted.

ANALOG INPUT LOW LEVEL OPERATION

Pads for external gain setting resistors (see Figure 7) have been provided so that the instrumentation amplifier can be user set for gains to 1000. The following formula can be used to calculate the value of the resistance: Gain = 100 $k\Omega / R_{EXT}$, where R_{EXT} is the resistance between pins 1 and 12 of the IA (R15, 94 in parallel form R_{EXT} in Figure 7). The gain adjustment potentiometer on the board will give an adjustment range of ±1%. Therefore, if an R_{EXT} with an accuracy of ±0.5% is used, the on-board potentiometer will have sufficient range for adjustment. Stable (50ppm) resistors should be used in this application. As shipped from the factory, R15 = 49.9kΩ. The settling time of the amplifier increases as the gain increases. A delay time of 41 microseconds is set at the factory to allow for multiplexer and amplifier settling times. This delay time is sufficient for amplifier gains of up to 50. For gains larger than 50, a longer delay time is required. A delay time of 81 microseconds will be obtained by removing R17. This delay time is sufficient for gains of up to 1000.

For lowest system noise, the ADC range should be set on the ±10V or 0 to 10V ranges with the amplifier providing all the system gain.

A 64/32 channel input board can be converted from single-ended operation to differential operation or vice versa by simply changing a few board jumpers (MP8632 from/to MP8664). Figure 5 indicates those jumpers that must be present for a given mode of operation. To convert from one mode to the other remove those jumpers indicated for the present type of operation and install those necessary for the desired mode of operation.

Required Jumpers for Differential	Required Jumpers for Single-ended
32 Channels W8 W9	64 Channels W6 W7 W10

FIGURE 5. MP8632 Channel Conversion

The differential mode of operation should be used for analog signals in noisy environments. The differential mode is particularly useful for low level signals since they are more prone to noise than high level signals. This board can also operate in a pseudo-differential mode. In this mode, the system has the number of channels of the single-ended mode, but the minus input of the IA is connected to a remote common rather than grounded on the board. This mode of operation is useful if there is a remote ground common to all the input signals. In this way the advantages of single-ended operation (maximum number of channels) and differential operation (better noise rejection) are combined. Jumpers W10 and W52 are installed and W7 is removed for this mode of operation.

ANALOG OUTPUT CHECKOUT

A static check of the two analog outputs is very simple. Load the L register with the output 1 data word and the H register with the output 2 data word. An SHLD instruction can then be used to transfer the data to the DAC's. The addresses of the analog outputs are set at the factory to values of $F700_{16}$ and $F701_{16}$. The ideal values for plus and minus full scale are shown in Figure 6.

DATA WORD	RANGE				
	±2.5V	±5.0V	±10V	0 to +5V	0 to +10
80_{16}	-2.500	-5.000	-10.000		
$7F_{16}$	+2.480	+4.961	+9.922		
00_{16}				0.000	0.000
FF_{16}				+4.980	+9.961

FIGURE 6. DAC Full Scale Values.

Two's complement coding is shown for bipolar ranges; straight binary for unipolar ranges.

If the SBC80/10 monitor is available, the Insert (I) and Substitute (S) commands can also be used to accomplish an output write.

To check the dynamic characteristics of the analog outputs the following program can be used.

```
        ORG   3C40H
        LXI   H, 0F700H ; Set H & L to DAC 1 Address
        MVI   A, 7FH  ;Initialize Data Word
LOOP:   MOV   M,A     ;Load DAC 1
        INX   H
        MOV   M, A    ;Load DAC 2
        DCX   H
        CMA           ;Compliment Data Word
        JMP   LOOP
        END
```

A 27 kHz square-wave will be present on both DAC outputs.

ANALOG INPUT CALIBRATION

These systems are set at the factory for a ±5V input range. If the input system range is to be changed, the following program may be used to adjust gain and offset.

```
REF     EQU   80H        ;Offset Ref = 80H, Full Scale Ref = 07FH
CO      EQU   01F8H      ;Monitor routines
CROUT   EQU   01F3H
NMOUT   EQU   02C2H
        ;
        ORG   3C50H
        ;
        LXI   H, 0F700H  ;Initialize
        LXI   SP, 3FFFH
BEG1:   MVI   E, 10H
BEG2:   LXI   B, 0
CLP:    MOV   A, M       ;Read data from board
        SUI   REF        ;Increment data count if data = REF
        JNZ   NEQ
        INR   B
NEQ:    INR   C          ;Have 100 conversions been made?
        MVI   A, 64H
        SUB   C
        JNZ   CLP
        MOV   A, B       ;Yes, Print data count
        CALL  NMOUT
        MVI   C, 20H     ;Print a space
        CALL  CO
        DCR   E          ;Full line been printed?
        JNZ   BEG2
        CALL  CROUT      ;Yes, Print CR & LF
        JMP   BEG1
        END
```

The program assumes that the system is under the control of the SBC80/10 prototype package monitor (M80P, version 1.0, March 1, 1976). It may be used for both offset and gain calibration. The system offset should be adjusted first, followed by the gain adjustment.

If the address of channel zero on the board has been changed from $F700_{16}$ then the LXI H instruction should reflect that change.

A G3C50 monitor command will begin program execution. After 100 conversions have been made, the value (in hex) of the B register will be printed. This value represents the number of times the data read from the board was equal to "REF" (80 for offset; 7F for gain).

RANGE	OFFSET	GAIN
±5V	-4.980	+4.941
0 to +10	+19.53mV	+9.941
0 to +5	+9.766mV	+4.971

FIGURE 8. Analog Input Calibration Values

Calibration is performed by connecting a voltage source capable of 0.01% accuracy to input channel zero (this could also be a DC voltage source of less absolute accuracy whose output is monitored by a 0.01% DVM).

The offset and gain adjustments are made while applying the voltage shown in Figure 8. For other ranges, the offset voltage adjustment is made at the most negative value of the range less one half least significant bit (LSB). An LSB is equal to the span (full scale range) divided by 256 for 8 bit resolution. The gain adjustment is made at the most positive value of the range less 1 1/2 LSB. Thus for a range of ±50mV, an LSB is $100mV/256 = 391\mu V$. The offset adjustment is made at $-50mV + 195\mu V = -49.80mV$ and the gain adjustment at $+50mV - 586\mu V = 49.41mV$. Before making these adjustments, however, the unit should be allowed to reach thermal equilibrium (about 30 minutes under power).

The offset adjustment is made first by using the appropriate offset calibration voltage. Run the calibration program and adjust the on board offset potentiometer until the B register contains a value between $1E_{16}$ and 46_{16} (30_{10} and 70_{10}).

To perform the gain adjustment change the data labeled "REF" in the calibration program from 80 to 7F, set the input voltage to the correct value as shown in Figure 8 and adjust the on board gain potentiometer in the same manner as described for offset.

If the SBC80 monitor is available, the substitute (S) command can be used to interrogate an input channel.

CONNECTOR PINOUT

P-3 ANALOG CONNECTOR PINOUT	Pin No.	Pin No.		P-4 ANALOG CONNECTOR PINOUT	Pin No.	Pin No.	
GND	1	2	GND	GND	1	2	GND
GND	3	4	GND	GND	3	4	GND
-15VDC	5	6	-15VDC	GND	5	6	GND
+15VDC	7	8	+15VDC	GND	7	8	GND
AO RET 1	9	10	AO RET 2	GND	9	10	GND
Analog Out 1	11	12	Analog Out 2	GND	11	12	GND
GND	13	14	GND	GND	13	14	GND
GND	15	16	GND	GND	15	16	GND
Remote Common	17	18	GND	GND	17	18	GND
IN0	19	20	IN32/RET0	IN16	19	20	IN48/RET16
IN1	21	22	IN33/RET1	IN17	21	22	IN49/RET17
IN2	23	24	IN34/RET2	IN18	23	24	IN50/RET18
IN3	25	26	IN35/RET3	IN19	25	26	IN51/RET19
IN4	27	28	IN36/RET4	IN20	27	28	IN52/RET20
IN5	29	30	IN37/RET5	IN21	29	30	IN53/RET21
IN6	31	32	IN38/RET6	IN22	31	32	IN54/RET22
IN7	33	34	IN39/RET7	IN23	33	34	IN55/RET23
IN8	35	36	IN40/RET8	IN24	35	36	IN56/RET24
IN9	37	38	IN41/RET9	IN25	37	38	IN57/RET25
IN10	39	40	IN42/RET10	IN26	39	40	IN58/RET26
IN11	41	42	IN43/RET11	IN27	41	42	IN59/RET27
IN12	43	44	IN44/RET12	IN28	43	44	IN60/RET28
IN13	45	46	IN45/RET13	IN29	45	46	IN61/RET29
IN14	47	48	IN46/RET14	IN30	47	48	IN62/RET30
IN15	49	50	IN47/RET15	IN31	49	50	IN63/RET31

8-BIT μP-COMPATIBLE D/A CONVERTER NE5018

NE5018-F,N

DESCRIPTION

The NE5018 is a complete 8-bit digital to analog converter subsystem on one monolithic chip. The data inputs have input latches, controlled by a latch enable pin. The data and latch enable inputs are ultra-low loading for easy interfacing with all logic systems. The latches appear transparent when the $\overline{LE}$ input is in the low state. When $\overline{LE}$ goes high, the input data present at the moment of transition is latched and retained until $\overline{LE}$ again goes low. This feature allows easy compatibility with most microprocessors.

The chip also comprises a stable voltage reference (5V nominal) and a high slew rate buffer amplifier. The voltage reference may be externally trimmed with a potentiometer for easy adjustment of full scale, while maintaining a low temperature co-efficient.

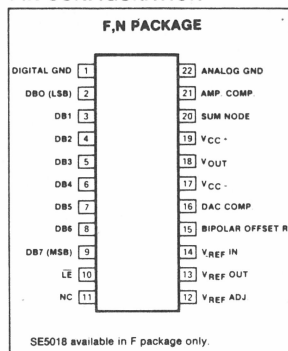
The output of the buffer amplifier may be offset so as to provide bipolar as well as unipolar operation.

FEATURES

- **8-bit resolution**
- **Input latches**
- **Low-loading data inputs**
- **On-chip voltage reference**
- **Output buffer amplifier**
- **Accurate to ± 1/2 LSB**
- **Monotonic to 8 bits**
- **Amplifier and reference both short-circuit protected**
- **Compatible with 2650, 8080 and many other μP's.**

APPLICATIONS

- **Precision 8-bit D/A converters**
- **A/D converters**
- **Programmable power supplies**
- **Test equipment**
- **Measuring instruments**
- **Analog-digital multiplication**

PIN CONFIGURATION

ABSOLUTE MAXIMUM RATINGS

	PARAMETER	RATING	UNIT
V_{CC+}	Positive supply voltage	18	V
V_{CC-}	Negative supply voltage	-18	V
V_{IN}	Logic input voltage	0 to 18	V
V_{REFIN}	Voltage at V_{REF} input	12	V
V_{REFADJ}	Voltage at V_{REF} adjust	0 to V_{REF}	V
V_{SUM}	Voltage at sum node	12	V
I_{REFSC}	Short-circuit current to ground at V_{REF} OUT	Continuous	
I_{OUTSC}	Short-circuit current to ground or either supply at V_{OUT}	Continuous	
I_{REF}	Reference input current	5	mA
P_D	Power dissipation*		
	-N package	800	mW
	-F package	1000	mW
T_A	Operating temperature range		
	SE5018	-55 to +125	°C
	NE5018	0 to +70	°C
T_{STG}	Storage temperature range	-65 to +150	°C
T_{SOLD}	Lead soldering temperature (10 seconds)	300	°C

*NOTE

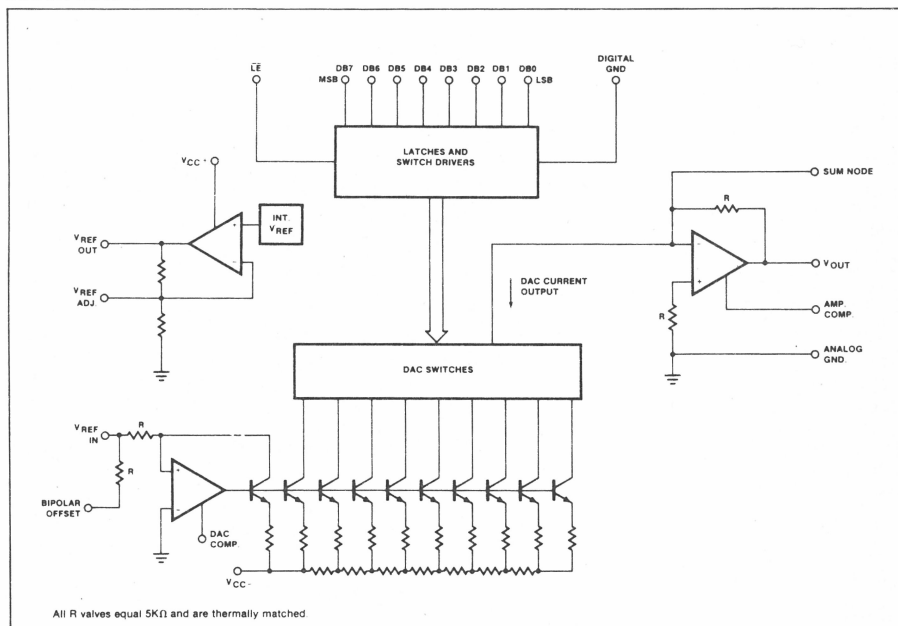
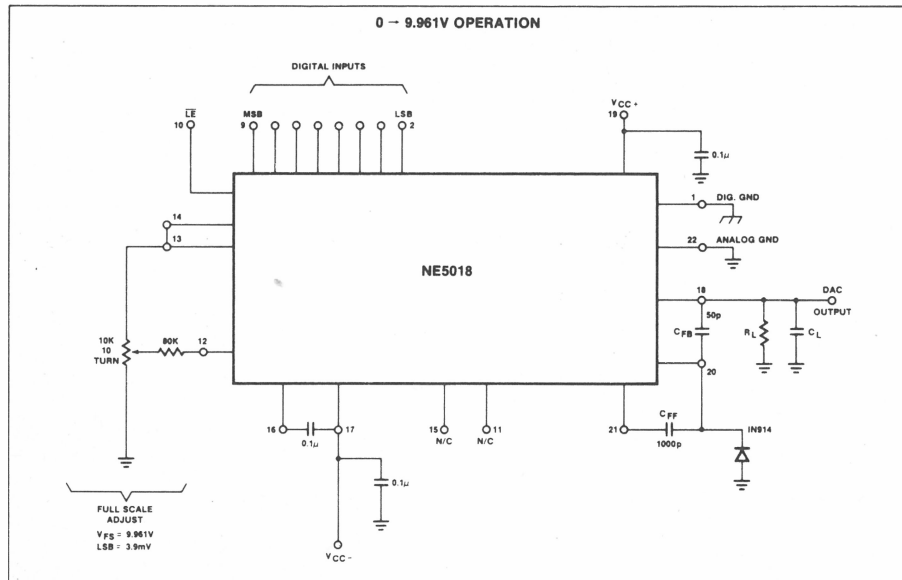
For N package, derate at 120°C/W above 35°C
For F package, derate at 75°C/W above 75°C

DC ELECTRICAL CHARACTERISTICS

$V_{CC}+ = 15V$, $V_{CC}- = -15V$, SE5018. $-55°C \leq T_A \leq 125°C$, NE5018. $0°C \leq T_A \leq 70°C$ unless otherwise specified.

	PARAMETER	TEST CONDITIONS	SE5018			NE5018			UNITS
			Min	Typ	Max	Min	Typ	Max	
$V_{CC}+$	Positive supply voltage			15			15		V
$V_{CC}-$	Negative supply voltage			-15			-15		V
	Resolution			8			8		bits
	Relative accuracy				±0.19			±0.19	%
T_S	Settling time	To ± 1/2LSB, 10V step		2			2		μs
PSRR	Power supply Rejection ratio	$V_{CC}+$ +12 to +18V $V_{CC}-$ -12 to -18V		±1			±1		mV/V
$I_{CC}+$	Positive supply current	$V_{CC}+ = 15V$		8			8		mA
$I_{CC}-$	Negative supply current	$V_{CC}- = -15V$		-10			-10		mA
$I_{IN}(0)$	Logic "0" input current	$V_{IN} = 0V$		5			5		μA
$V_{IN}(0)$	Logic "0" input voltage				0.8			0.8	V
$V_{IN}(1)$	Logic "1" input voltage		2.0			2.0			V
T_{PWLE}	Latch enable pulse width			400			400		ns

BLOCK DIAGRAM

All R valves equal 5KΩ and are thermally matched.

EQUIVALENT SCHEMATIC

Signetics Corporation
PO Box 9052
811 East Arques Avenue
Sunnyvale California 94086
Telephone 408/739-7700

MP10, MP11

MICROPROCESSOR ANALOG OUTPUT COMPONENTS

COMPATIBLE WITH:

8080 (Intel)
9080A (AMD)
Z-80 (Zilog)
6800 (Motorola)
8008 (Intel)
F-8 (Fairchild)
SC/MP (National)
650X (MOS Technology)
9002 (Electronic Arrays)

USE AS ANALOG INPUT AND OUTPUT

EASY TO USE
- Completely compatible with most Microprocessors
- No external logic required
- Timing compatible
- Memory mapped

SAVES DEVELOPMENT MONEY AND TIME

COMPLETELY SELF-CONTAINED

International Airport Industrial Park • P.O. Box 11400 • Tucson, Arizona 85734
Tel: 602-294-1431 • Twx: 910-952-1111 • Cable: BBRCORP • Telex: 66-6491

DESCRIPTION

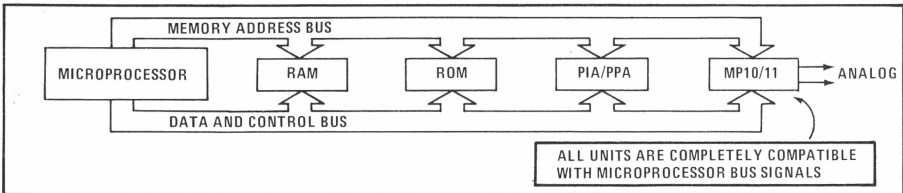
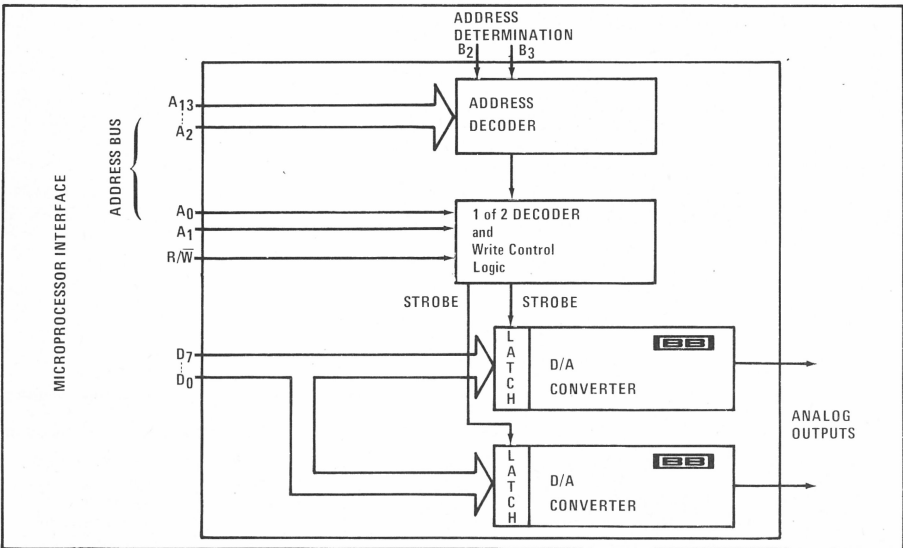
These microprocessor peripherals provide an analog interface compatible with most microprocessors. The MP10 and MP11 are electrically and functionally microprocessor compatible in static or dynamic situations.

These units are complete analog systems packaged in 32 pin triple wide dual-in-line packages. They contain two 8 bit D/A converters which are internally trimmed for gain and offset so that no external trimming is required. All necessary interface, timing and address decoding logic is also included.

The MP10 is designed to be used with 8080A and 8008 type microprocessors. It can be used with SC/MP if pull-up resistors are added to the address bus, with the F-8 Dynamic or Static memory interface chip if the RAM WRITE signal is a minimum of 430nsec and with the Z-80 if $t_w(\phi H) = t_w(\phi L) = 500$ns. The MP11 is designed to be used with 6800, 650X and 9002 type microprocessors.

The address lines A_2 through A_{13}, B_2 and B_3 of the MP10 are CMOS compatible so that they can be directly connected to the address bus of an 8080 or 8008. All other input lines require standard TTL voltages. The address lines A_2 through A_{13} and B_2 of the MP11 are LSTTL compatible so they can be directly connected to the address bus of a 6800 or 650X. All other input lines require standard TTL voltages but are high impedance requiring only microamp drive currents.

THEORY OF OPERATION

When programming these peripherals, the user treats them as memory. Because the D/A converter input is an 8 bit word, one 8 bit memory location is required for each channel. Since these units are treated as memory, a single instruction is all that's needed to write to an output channel. For instance, when the MP10 is used with an 8080, a single instruction, SHLD, can be used to output data to both D/A converter channels from the H and L register pair. Likewise, when the MP11 is used with the 6800 or 650X, a single STX instruction can be used to output data to both D/A converter channels from the index register. The MP10 and the MP11 require an initialization as would any programmable peripheral.

MP10, MP11 BLOCK DIAGRAM

ELECTRICAL SPECIFICATIONS

(Typical at 25°C and rated supplies unless otherwise noted.)

	MP10/MP11
ANALOG OUTPUT	
Number of analog outputs	2
Output voltage range	±10V
Output impedance	1Ω
Output settling time	25 μsec
TRANSFER CHARACTERISTICS	
Resolution	8 bit binary (complementary binary)
One LSB	78.1mV
Throughput accuracy (max)	±0.4% FSR[1,2]
Temperature coefficient of accuracy	±0.008% FSR/°C

	MP10/MP11
DIGITAL INPUT/OUTPUT	
All signals compatible with the microprocessor bus	
An analog output channel selected by:	A0
Input data bits read by:	D0 - D7
POWER REQUIREMENTS	
	+5VDC ±5% at 90 mA
	+15V ±3% at 30 mA
	-15V ±3% at 30 mA
TEMPERATURE RANGE	
Operating temperature range	0 - 70°C
Storage temperature range	-55°C to +85°C

1. FSR is Full Scale Range = 20V.
2. Accuracy components are: Linearity Error = ±0.2% FSR; Gain Error = ±0.1% FSR, Offset Error = ±0.1% FSR.

MECHANICAL SPECIFICATIONS

MATERIAL: Alumina
WEIGHT: 14 grams (0.5 oz)
PINS: Pin material and plating composition conform to Method 2003 (solderability) of Mil-Std-883 (except paragraph 3.2).
MATING CONNECTOR: 2302MC

PIN CONNECTIONS

MP10

8080 Pin Connections	Pin	Function	Function	Pin	8080 Pin Connections
1	1	A10	A11	32	40
2	2	Common	A13	31	38
3	3	D4	A12	30	37
4	4	D5	A 9	29	35
5	5	D6	A 8	28	34
6	6	D7	A 7	27	33
7	7	D3	A 6	26	32
8	8	D2	A 5	25	31
9	9	D1	A 4	24	30
10	10	D0	A 3	23	29
12	11	Reset	A 2	22	27
18	12	R/$\overline{W}$	B 2	21	—
26	13	A1	B 3	20	—
25	14	A0	+5V	19	20
—	15	+15V	Out 1	18	—
—	16	-15V	Out 2	17	—

MP11

6800 Pin Connections	Pin	Function	Function	Pin	6800 Pin Connections
—	1	Output 1	-15V	32	—
—	2	Output 2	+15V	31	—
8	3	+5V	R/$\overline{W}$	30	34
37	4	Enable	$\overline{Reset}$	29	40
9	5	A0	D0	28	33
10	6	A1	D1	27	32
11	7	A2	D2	26	31
12	8	A3	D3	25	30
13	9	A4	D4	24	29
14	10	A5	D5	23	28
15	11	A6	D6	22	27
16	12	A7	D7	21	26
17	13	A8	Common	20	21
18	14	A9	B2	19	—
19	15	A10	A13	18	23
20	16	A11	A12	17	22

Symbol	Min	Max	Units
T_{WP}	430	—	ns
T_{DW}	50	—	ns
T_{WD}	65	—	ns
T_{AW}	620	—	ns
T_{WA}	35	—	ns
T_{WD}	65	—	ns
T_{WC}	35	—	ns
T_{WB}	—	635	ns
T_{AO}	16	25	µs
T_{AC}	—	600	ns

FIGURE 1. MP10 Timing Diagram.

Symbol	Min	Max	Unit
T_E	0.450	25	µs
T_{AEW}	180	—	ns
T_{DSU}	300	—	ns
T_{WE}	130	—	ns
T_{HW}	10	—	ns
T_{PDW}	—	1.0	µs
T_{AO}	17	25	µs

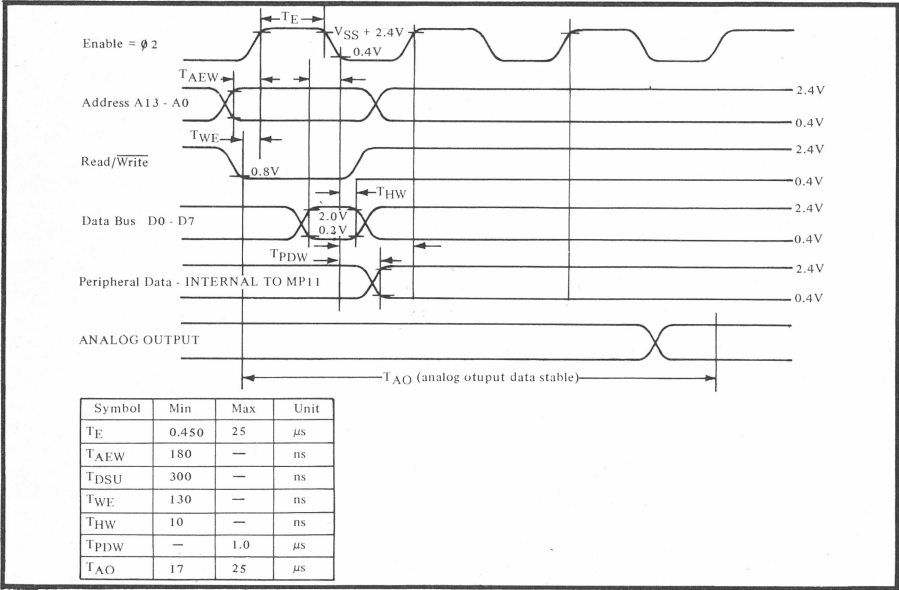



FIGURE 2. MP11 Timing Diagram.

PROGRAMMING

These units are easily programmed since all are treated as memory locations. They use any memory reference instruction that can write data from internal registers or the accumulator. A single instruction can be used to write data to one or both channels. When the MP10 is used with an 8080, a single SHLD instruction referenced to the lower of the two addresses will automatically transfer the data in the H register to DAC1 and the data in the L register to DAC2. An STA instruction will transfer the data in the accumulator to either DAC. When the MP11 is used with a 6800, a single STX instruction referenced to the lower of the two addresses will automatically transfer the eight upper bits of the index register to DAC1 and the eight lower bits to DAC2. An STAA instruction will transfer the contents of the accumulator to either DAC. Of course, if direct addressing is not desired, MOV instructions may be used to transfer data from internal registers to a specific DAC memory location. As with any programmable peripheral, the MP10 and MP11 must be initialized.

MP10 INITIALIZATION

The RESET input controls the status of the control register of the MP10. An active high on this line will reset the control register to all "zeros".

The MP10 will require initialization every time RESET is activated. If RESET is connected to ground, the MP10 must be initialized only once before output of the data.

MP10 INITIALIZATION SEQUENCE:

1. Load initialization address
2. Load initialization data

MP10 INITIALIZATION ADDRESS:

A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	A_9	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0
X	X	1	1	1	1	1	1	1	1	1	1	a	a	1	1
												User Defined			

X = don't care, not connected to MP10
1 = True

MP10 INITIALIZATION DATA

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	
1	0	0	0	0	0	0	0	$= 80_{16}$

For 8080 the sequence may look as follows:

LXI H, ADDR; ADDR = Initialization address
Loads H & L registers with initialization address

MVI M, DATA; DATA = 80
Loads initialization data (80_{16}) to initialization address

The initialization sequence assigns internal registers to function as input registers for the D/A converters. Now data can be written into the MP10. This is accomplished by outputing the correct MP10 address:

The B_2 and B_3 inputs determine the address to which the MP10 will respond. The four memory locations which are possible are outlined below:

B_2	B_3	A_2	A_3
0	0	0	0
0	1	0	1
1	0	1	0
1	1	1	1

At the time that the address appears on the address bus, data will appear on the data bus and a R/$\overline{W}$ pulse will be generated by the microprocessor. 25μs later, the analog voltage will be stable at the selected output. Timing requirements shown in Figure 1 must be satisfied in order for the MP10 to be initialized and operate correctly. These timing requirements are completely compatible with the 8080.

MP11 INITIALIZATION

The $\overline{RESET}$ input controls the status of the control and peripheral registers of the MP11. The initialization sequence will differ if $\overline{RESET}$ is connected to a master reset line of a microprocessor or if it is hard-wired to V_{cc}. The MP11 will require initialization every time the $\overline{RESET}$ line is activated low. If the $\overline{RESET}$ line is hard wired to V_{cc}, the MP11 must be initialized only once before output of the data is attempted.

MP11 ADDRESS STRUCTURE

A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	A_9	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0
X	X	1	1	1	1	1	1	1	1	1	1	0	a	Y	Y

A_{15}, A_{14} - don't care, not connected to MP11
A_2 - Address is user selectable
A_0, A_1 - Addresses control the initialization sequence

Initialization sequence when $\overline{RESET}$ is hard wired to V_{cc}:

1. Load accumulator with "zeros"
2. Store accumulator at memory locations:

A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	A_9	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	
X	X	1	1	1	1	1	1	1	1	1	1	0	a	1	0	Address of Control register A

A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	A_9	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	
X	X	1	1	1	1	1	1	1	1	1	1	0	a	1	1	Address of Control register B

3. Load accumulator with "ones"
4. Store accumulator at memory locations:

A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	A_9	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	
X	X	1	1	1	1	1	1	1	1	1	1	0	a	0	0	Address of Peripheral register A
X	X	1	1	1	1	1	1	1	1	1	1	0	a	0	1	Address of Peripheral register B
X	X	1	1	1	1	1	1	1	1	1	1	0	a	1	0	Address of Control register A
X	X	1	1	1	1	1	1	1	1	1	1	0	a	1	1	Address of Control register B

For the 6800 this sequence can be written as follows:

```
LDAA    "zeros"                            Loads Zeros in accumulator
STAA    Address of control register A      Stores zero's in C.R.A.
STAA    Address of control register B      Stores zero's in C.R. B
LDAA    "ones"                             Loads one's in accumulator
STAA    Address of peripheral register A   Stores one's in P.R.A
STAA    Address of peripheral register B   Stores one's in C.R.A
STAA    Address of control register B      Stores one's in C.R.B

LDX     # $0000                            Loads zero's in index register
STX     $ Address control register A       Stores zero's in C.R. A and B
LDX     # $1111                            Loads one's in index register
STX     $ Address peripheral register A    Stores one's in P.R. A and B
STX     $ Address control register A       Stores one's in C.R. A and B
```

Initialization sequence when $\overline{RESET}$ line is connected to master reset (control registers A and B are always set to zero after master reset and only ones need to be stored in the registers):

```
LDAA    "ones"
STAA    Address Peripheral register A
STAA    Address Peripheral register B
STAA    Address Control register A
STAA    Address Control register B
```

or as:

```
LDXX    # $1111
STX     $ Address Peripheral register A
STX     $ Address Control register A
```

Now data can be written into MP11. This is accomplished by outputing the correct MP11 address:

A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	A_9	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	
X	X	1	1	1	1	1	1	1	1	1	1	0	a	0	0	OUTPUT 1
X	X	1	1	1	1	1	1	1	1	1	1	0	a	0	1	OUTPUT 2

At the time that the address appears on the address bus, data will appear on the data bus, and if the R/W and Enable pulses are correctly timed, 25μs from the true address the analog voltage will be stable at the selected output.

Timing requirements shown in Figure 1 must be satisfied for the MP11 to be initialized and operate correctly. All timing requirements are completely compatible with 6800 microprocessors. User definable address line A_2 used in conjunction with the B_2 input allows the user to place the MP11 in two different memory locations or use two different MP11's in order to expand the analog system to four outputs. When B_2 is wired to logical 1, the MP11 responds to an A_2 address of 0 and when B_2 is wired to a logical 0, the MP11 responds to an A_2 address of 1.

TEST PROGRAMS

The test circuit and test programs following allow the user to test the operation of the MP10 or MP11. The test may be conducted by setting up the MP10/MP11 as shown in Figure 3. The microprocessor system should have a teletype/CRT terminal interface. The programs will step through several output voltage levels for each DAC output (see Figure 4). Notice how the software is different for the two test programs to illustrate two software approaches.

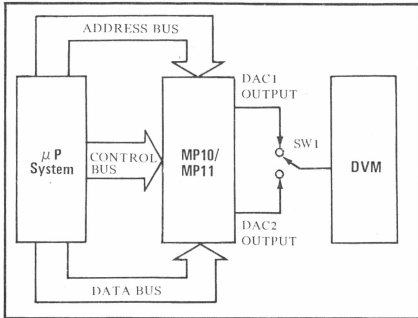



FIGURE 3. Test Circuit for MP10/MP11.

MP10 Test Program

```
                Initialize MP10
          LXI H ADDR X     Address of the first
                           byte of data.
LOOP 1  ┌→MOV A, M         Load ACC with first byte
        │                  of data.
        │ STA ADDR2        Output to MP10 DAC1
        │ INX H            Increment ADDR1
        │ CALL CI   )      Call Input routine
        │ CPI       }      Wait for any character
        │ 8D        )      except carriage return
        └─JNZ LOOP1
          LXI ADDR X
LOOP 2  ┌→MOV A, M
        │ STA ADDR3        Output to MP10 DAC2
        │ INX H            Increment ADDR1
        │ CALL CI   )
        │ CPI       }      Wait for any character
        │ 8D        )      except carriage return
        └─JNZ LOOP2
          RET
```

The MP10 test program will output five different voltages from DAC1 and then from DAC2 (see Figure 4). DAC1 will initially output -10V. To step through the other values for DAC1 enter any character other than carriage return (CR). To transfer control to DAC2, enter CR. DAC2 will output -10V. To step through the other values for DAC2 enter any character except CR. To exit the test program, enter CR.

Store the following codes in memory beginning with location ADDR X:

```
ADDR X     ← FF
ADDR X + 1 ← BF
ADDR X + 2 ← 7F
ADDR X + 3 ← 3F
ADDR X + 4 ← 00
```

ADDR 2 is the address of output 1, ADDR 3 is the address of output 2:

MP11 Test Program

```
                                  Initialize MP11
       LDX   # $ FFFF;            Load index register
       STX   ADDR 1;              Store FF in each DAC
       JSR   INP
       LDX   # $ BFBF;            Load index register
       STX   ADDR 1;              Store BF in each DAC
       JSR   INP
       LDX   # $ 7F7F;            Load index register
       STX   ADDR 1;              Store 7F in each DAC
       JSR   INP
       LDX   # $ 3F3F;            Load index register
       STX   ADDR 1;              Store 3F in each DAC
       JSR   INP
       LDX   # $ 0000;            Load index register
       STX   ADDR 1;              Store 00 in each DAC
       JSR   INP
INP    LDAA    ADDR X             Load Status }  Wait for
                                  of ACIA     }  TTY
       Bit A  #01                             }  input
       BEQ   INP
       LDA A   ADDR X + 1         Load Data   }
                                  From ACIA   }
       CMP   A                                }  Jump back
       8D                                     }  to test
       BNE   Back                             }  program or
       JMP   Return                           }  return to
BACK   RTS                                    }  main program
```

The MP11 test program will output -10V from both DAC1 and DAC2 then wait for an input from the TTY. Any character except CR will advance both DAC's of the MP11 to the next value as defined in Figure 4. CR terminates test program by jumping to RETURN.

ADDR 1 is the address of output 1, ADDR X is the address of the ACIA.

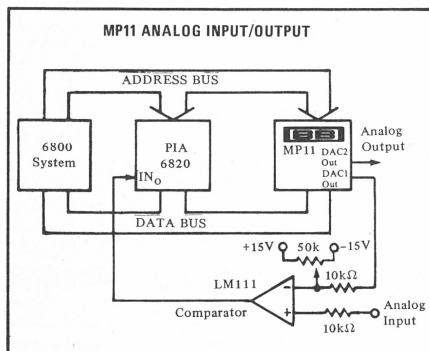
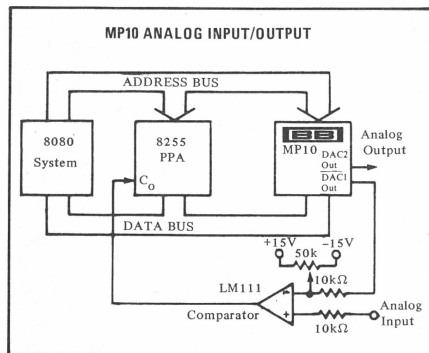
Step	Ideal Output	Actual Output Limits
1	-10V	-9.922V to -10.078V
2	-5.0V	-4.922 to -5.078
3	0.000V	-0.078 to +0.078
4	+5.0V	+4.972 to +5.078
5	+9.922V	+9.844 to +10.000

FIGURE 4. Output Voltages for Test Programs.

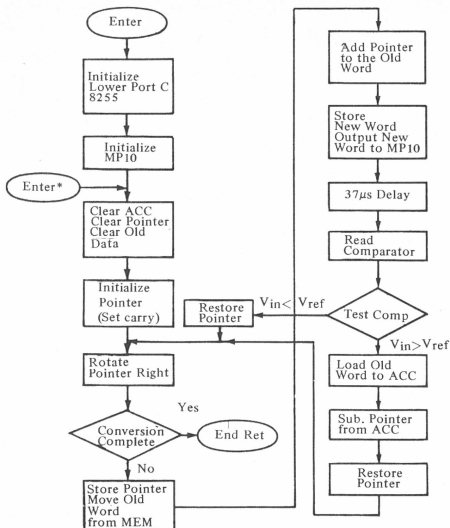
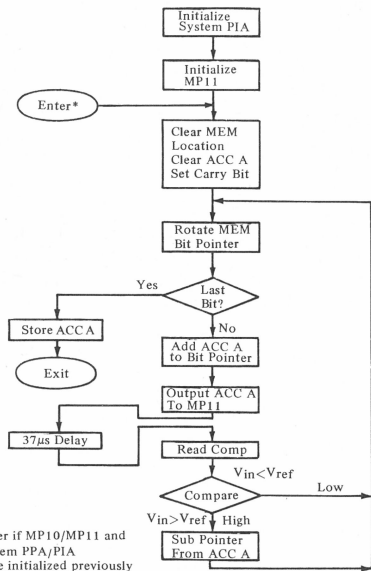
APPLICATIONS

ANALOG INPUT AND ANALOG OUTPUT

Although the MP10 and MP11 are analog output peripherals, they can be easily adapted to provide both analog inputs and outputs.

With the addition of a few external components, these units can each provide one analog input and one analog output for your system as shown below:

These systems use the microcomputer system to perform the logic of a successive approximation A/D converter, using one channel of the MP10 or MP11 to provide the D/A converter reference function required. In a successive approximation converter, the analog input is compared to known outputs of a D/A converter. First, the microcomputer turns the MSB on, waits for the settling time of the MP10 or MP11, and the switching time of the comparator, then reads the status. If the comparator indicates that the MSB voltage is smaller than the analog input, the MSB input to the MP10/MP11 stays "on" and the next most significant bit is turned on. If the comparator indicates that the MSB value is larger than the analog input, the microcomputer will turn the MSB "off" and turn "on" the next most significant bit. In this way all 8 bits of the D/A converter are tested. When the conversion is complete, the input of the D/A converter will be a digital representation of the analog input. This value will also be stored in the microprocessor's accumulator (complementary binary).

The A/D conversion will require approximately 900 microseconds when performed in this manner. Burr-Brown will shortly have available a detailed application note describing this process including all software required.

FLOWCHART USING 8080 and MP10

FLOWCHART USING 6800 and MP11

* Enter if MP10/MP11 and system PPA/PIA were initialized previously

SEPTEMBER 1977

ADC0816/ADC0817 Single Chip Data Acquisition System

General Description

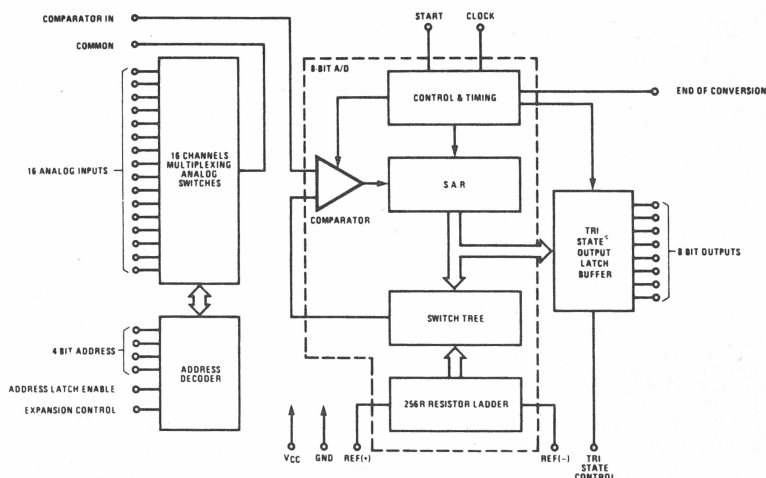
The ADC0816, ADC0817 (MM74C948) data acquisition components are monolithic CMOS devices with an 8-bit analog-to-digital converter, a 16-channel multiplexer and microprocessor compatible control logic. The 8-bit A/D converter uses successive approximation as the conversion technique. The converter features a high impedance chopper stabilized comparator, a 256R voltage divider with analog switch tree and a successive approximation register. The 16-channel multiplexer can directly access any one of 16 single-ended analog signals and provides the logic for additional channel expansion. Signal conditioning of any analog input signal is eased by direct access to the input of the 8-bit A/D converter.

The device eliminates the need for external zero and full-scale adjustments and features an absolute accuracy $\leq$ 1 LSB including quantitizing error. Easy interfacing to microprocessors is provided by the latched and decoded address inputs and latched TTL TRI-STATE® outputs.

The design of the ADC0816, ADC0817 has been optimized by incorporating the most desirable aspects of several A/D conversion techniques. The ADC0816, ADC0817 offers high speed, high accuracy, minimal temperature dependence, excellent long-term accuracy and repeatability, and consumes minimal power. These features make this device ideally suited to applications such as process control, industrial control, and machine control.

Features

- Total unadjusted error $< \pm 1/2$ LSB
- Linearity error $< \pm 1/2$ LSB
- No missing codes
- Guaranteed monotonicity
- No offset adjust required
- No scale adjust required
- Conversion time of 100 μs
- Easy microprocessor interface
- Latched TRI-STATE output
- Latched address input
- Ratiometric conversion
- Single 5V supply
- Low power consumption—15 mW

Block Diagram

Absolute Maximum Ratings (Notes 1 and 2)

Voltage at Any Pin Except Control Inputs	−0.3V to V_{CC} + 0.3V
Voltage at Control Inputs	−0.3V to + 15V
(Start, TRI-STATE, Clock, ALE, ADD A, ADD B, ADD C, ADD D, Expansion Control)	
Operating Temperature Range	−40°C to +85°C
Storage Temperature Range	−65°C to +150°C
Package Dissipation (at 25°C)	500 mW
Operating V_{CC} Range	4.5V to 6V
Absolute Maximum V_{CC}	6.5V
Lead Temperature (Soldering, 10 seconds)	300°C

DC Electrical Characteristics

$4.75V \leq V_{CC} \leq 5.25V$, $-40°C \leq T_A \leq +85°C$ unless otherwise noted, (Note 2)

	PARAMETER	CONDITIONS	MIN	TYP	MAX	UNITS
$V_{IN(1)}$	Logical "1" Input Voltage	V_{CC} = 5V	V_{CC}−1.5			V
$V_{IN(0)}$	Logical "0" Input Voltage	V_{CC} = 5V			1.5	V
$V_{OUT(1)}$	Logical "1" Output Voltage	I_O = −360 µA	V_{CC}−0.4			V
$V_{OUT(0)}$	Logical "0" Output Voltage	I_O = 1.6 mA			0.45	V
$V_{OUT(0)}$	Logical "0" Output Voltage EOC	I_O = 1.2 mA			0.45	V
$I_{IN(1)}$	Logical "1" Input Current (The Control Inputs)	V_{IN} = 15V			1.0	µA
$I_{IN(0)}$	Logical "0" Input Current (The Control Inputs)	V_{IN} = 0	−1.0			µA
I_{CC}	Supply Current	Clock Frequency = 500 kHz		300	1000	µA
I_{OUT}	TRI-STATE Output Current	V_O = 5V			3	µA
		V_O = 0	−3			µA

Note 1: "Absolute Maximum Ratings" are those values beyond which the safety of the device cannot be guaranteed. Except for "Operating Temperature Range" they are not meant to imply that the devices should be operated at these limits. The table of "Electrical Characteristics" provides conditions for actual device operation.

Note 2: All voltages measured with respect to GND unless otherwise specified.

Note 3: Non-linearity error is the maximum deviation from a straight line through the end points of the A/D transfer characteristic, *(Figure 2)*.

Note 4: Zero error is the difference between the output of an ideal and the actual A/D for zero input voltage, *(Figure 2)*.

Note 5: Full-scale error is the difference between the output of an ideal and the actual A/D for full-scale input voltage, *(Figure 2)*.

Note 6: Total unadjusted error is the maximum sum of non-linearity, zero and full-scale errors, *(Figure 3)*.

Note 7: Quantization error is the ±1/2 LSB uncertainty caused by the converter's finite resolution, *(Figure 3)*.

Note 8: Absolute Accuracy describes the difference between the actual input voltage and the full-scale weighted equivalent of the binary output code; included are quantizing and all other errors. Although rarely provided on data sheets, it is the best indication of a converter's true performance, *(Figure 3)*.

Note 9: Supply rejection relates to the ability of an ADC to maintain accuracy as the supply voltage varies. The supply and $V_{REF(+)}$ are varied together and the change in accuracy is measured with respect to full-scale.

Note 10: Comparator input current is a bias current into or out of the chopper stabilized comparator. The bias current varies directly with clock frequency and has little temperature dependence, *(Figure 5)*.

DC Electrical Characteristics (Continued)

ANALOG MULTIPLEXER

ADC0816, ADC0817 $-40^\circ C \leq T_A \leq +85^\circ C$ unless otherwise noted

	PARAMETER	CONDITIONS	MIN	TYP	MAX	UNITS
R_{ON}	Analog Multiplexer ON Resistance	(Any Selected Channel) $T_A = 25^\circ C$, $R_L = 10k$		1.5	3 6	kΩ kΩ
ΔR_{ON}	Δ ON Resistance Between Any 2 Channels	(Any Selected Channel) $R_L = 10k$		75		Ω
$I_{OFF(+)}$	OFF Channel Leakage Current	$V_{CC} = 5V$, $V_{IN} = 5V$, $T_A = 25^\circ C$		10	200	nA
$I_{OFF(-)}$	OFF Channel Leakage Current	$V_{CC} = 5V$, $V_{IN} = 0$, $T_A = 25^\circ C$	−200	−10		nA

CONVERTER SECTION $V_{CC} = V_{REF(+)} = 5V$, $V_{REF(-)} = GND$, $V_{IN} = V_{COMPARATOR\ IN}$, $f_c = 640\ kHz$

ADC0816CCN $-40^\circ C \leq T_A \leq +85^\circ C$ unless otherwise noted

PARAMETER	CONDITIONS	MIN	TYP	MAX	UNITS
Resolution		8			Bits
Non-Linearity	(Note 3)		±1/4	±1/2	LSB
Zero Error	(Note 4)		±1/4	±1/2	LSB
Full-Scale Error	(Note 5)		±1/4	±1/2	LSB
Total Unadjusted Error	$T_A = 25^\circ C$		±1/4	±1/2	LSB
	(Note 6)		±1/4	±3/4	LSB
Quantization Error	(Note 7)			±1/2	LSB
Absolute Accuracy	$T_A = 25^\circ C$		±3/4	±1	LSB
	(Note 8)		±3/4	±1 1/4	LSB

ADC0817CCN $T_A = 25^\circ C$

PARAMETER	CONDITIONS	MIN	TYP	MAX	UNITS
Resolution		8			Bits
Non-Linearity	(Note 3)		±1/2	±1	LSB
Zero Error	(Note 4)		±1/4	±1/2	LSB
Full-Scale Error	(Note 5)		±1/4	±1/2	LSB
Total Unadjusted Error	(Note 6)		±1/2	±1	LSB
Quantization Error	(Note 7)			±1/2	LSB
Absolute Accuracy	(Note 8)		±1	±1 1/2	LSB

ADC0816CCN $-40^\circ C \leq T_A \leq +85^\circ C$
ADC0817CCN $T_A = 25^\circ C$

PARAMETER	CONDITIONS	MIN	TYP	MAX	UNITS
Power Supply Rejection	$4.75V \leq V_{CC} = V_{REF(+)} \leq 5.25V$, (Note 9)		0.05	0.15	%/V
Comparator Input Current	$f_c = 640\ kHz$, (Note 10)	−2	±0.5	2	μA
Ladder Resistance	From Ref(+) to Ref(−)	1	4.5		kΩ

DC Electrical Characteristics (Continued)

DESIGN GUIDELINES

ADC0816CCN, ADC0817CCN

PARAMETER		CONDITIONS	MIN	TYP	MAX	UNITS
V_{LAD}	Voltage Across Ladder	From Ref(+) to Ref(−)	0.512	5.12	5.25	V
$V_{REF(+)}$	Voltage, Top of Ladder	Measured at Ref(+)		V_{CC}	V_{CC}+0.1	V
$\frac{V_{REF(+)} + V_{REF(-)}}{2}$	Voltage, Center of Ladder	Measured at $R_{LADDER}/2$	$\frac{V_{CC}}{2}-0.1$	$\frac{V_{CC}}{2}$	$\frac{V_{CC}}{2}+0.1$	V
$V_{REF(-)}$	Voltage, Bottom of Ladder	Measured at Ref(−)	−0.1	0		V

AC Electrical Characteristics

ADC0816CCN, ADC0817CCN $T_A = 25°C$, $V_{CC} = V_{REF(+)} = 5V$, $V_{REF(-)} = GND$

PARAMETER		CONDITIONS	MIN	TYP	MAX	UNITS
t_{WS}	Start Pulse Width	*(Figure 5)*	200	100		ns
t_{WALE}	Minimum ALE Pulse Width	*(Figure 5)*	200	100		ns
t_s	Address Set-Up Time	*(Figure 5)*	50	25		ns
t_H	Address Hold Time	*(Figure 5)*	50	25		ns
t_D	Analog MUX Delay Time From ALE	Common Tied to Comparator In, $R_S + R_{ON} \leq 5\ k\Omega$, $C_L = 10\ pF$		1	2.5	μs
t_{H1}, t_{H0}	TRI-STATE Control to Q Logic State	$C_L = 50\ pF$		125	250	ns
t_{1H}, t_{0H}	TRI-STATE Control to Hi-Z	$C_L = 10\ pF$, $R_L = 10k$		125	250	ns
t_c	Conversion Time	$f_c = 640\ kHz$, *(Figure 5)*	90	100	114	μs
f_c	Clock Frequency		10	640	1200	kHz
t_{EOC}	EOC Delay Time	*(Figure 5)*	1		8	Clock Periods
C_{IN}	Input Capacitance	At Control Inputs		10	15	pF
		At MUX Inputs		5	7.5	pF
C_{OUT}	TRI-STATE Output Capacitance	At TRI-STATE Outputs, (Note 11)		5	7.5	pF

Note 11: Capacitance guaranteed by periodic testing.

Timing Diagram

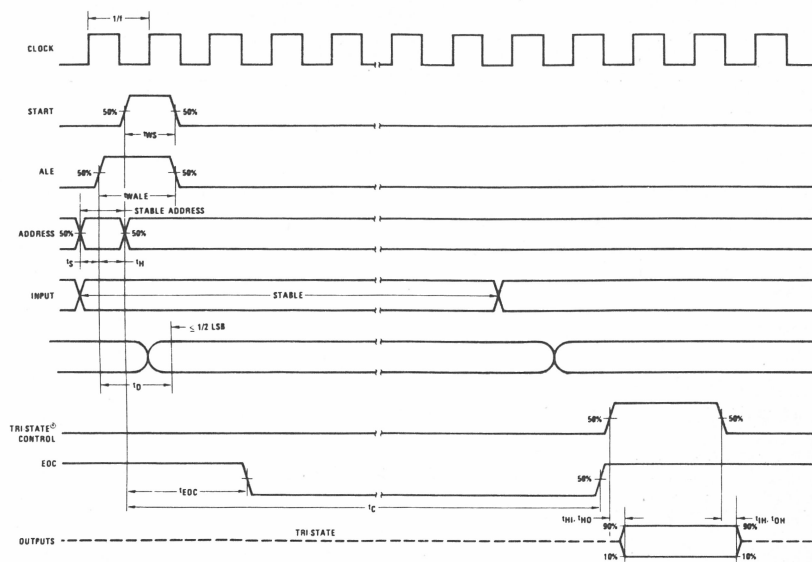



FIGURE 5

Typical Performance Characteristics

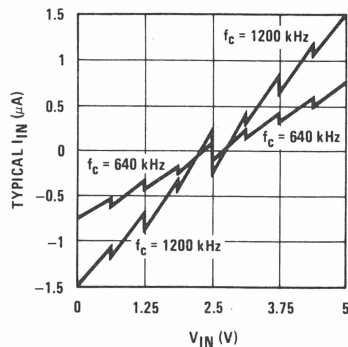



FIGURE 6. Comparator I_{IN} vs V_{IN} ($V_{CC} = V_{REF} = 5V$)

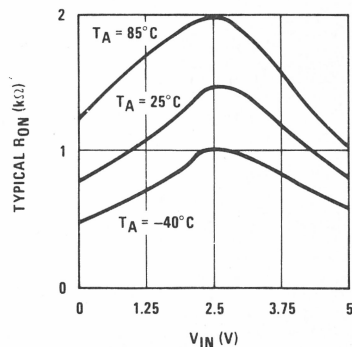



FIGURE 7. Multiplexer R_{ON} vs V_{IN} ($V_{CC} = V_{REF} = 5V$)

Functional Description

Multiplexer: The device contains a 16-channel single-ended analog signal multiplexer. A particular input channel is selected by using the address decoder. Table I shows the input states for the address line and the expansion control line to select any channel. The address is latched into the decoder on the low-to-high transition of the address latch enable signal.

Additional single-ended analog signals can be multiplexed to the A/D converter by disabling all the multiplexer inputs. The additional external signals are connected to the comparator input and the device ground. Additional signal conditioning (i.e., prescaling, sample and hold, instrumentation amplification, etc.) may also be added between the analog input signal and the comparator input.

TABLE I

SELECTED ANALOG CHANNEL	ADDRESS LINE				EXPANSION CONTROL
	D	C	B	A	
IN0	L	L	L	L	H
IN1	L	L	L	H	H
IN2	L	L	H	L	H
IN3	L	L	H	H	H
IN4	L	H	L	L	H
IN5	L	H	L	H	H
IN6	L	H	H	L	H
IN7	L	H	H	H	H
IN8	H	L	L	L	H
IN9	H	L	L	H	H
IN10	H	L	H	L	H
IN11	H	L	H	H	H
IN12	H	H	L	L	H
IN13	H	H	L	H	H
IN14	H	H	H	L	H
IN15	H	H	H	H	H
All Channels OFF	X	X	X	X	L

X = don't care

CONVERTER CHARACTERISTICS

The Converter

The heart of this single chip data acquisition system is its 8-bit analog-to-digital converter. The converter is designed to give fast, accurate, and repeatable conversions over a wide range of temperatures. The converter is partitioned into 3 major sections: the 256R ladder network, the successive approximation register, and the comparator. The converter's digital outputs are positive true.

The 256R ladder network approach *(Figure 1)* was chosen over the conventional R/2R ladder because of its inherent monotonicity, which guarantees no missing digital codes. Monotonicity is particularly important in closed loop feedback control systems. A non-monotonic relationship can cause oscillations that will be catastrophic for the system. Additionally, the 256R network does not cause load variations on the reference voltage.

The bottom resistor and the top resistor of the ladder network in *Figure 1* are not the same value as the remainder of the network. The difference in these resistors causes the output characteristic to be symmetrical with the zero and full-scale points of the transfer curve. The first output transition occurs when the analog signal has reached +1/2 LSB and succeeding output transitions occur every 1 LSB later up to full-scale.

The successive approximation register (SAR) performs 8 iterations to approximate the input voltage. For any SAR type converter, n-iterations are required for an n-bit converter. *Figure 2* shows a typical example of a 3-bit converter. In the ADC0816, ADC0817, the approximation technique is extended to 8 bits using the 256R network.

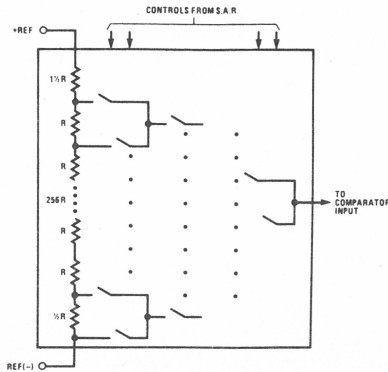



FIGURE 1. Resistor Ladder and Switch Tree

Functional Description (Continued)

The A/D converter's successive approximation register (SAR) is reset on the positive edge of the start conversion (SC) pulse. The conversion is begun on the falling edge of the start conversion pulse. A conversion in process will be interrupted by receipt of a new start conversion pulse. Continuous conversion may be accomplished by tying the end-of-conversion (EOC) output to the SC input. If used in this mode, an external start conversion pulse should be applied after power up. End-of-conversion will go low between 1 and 8 clock pulses after the rising edge of start conversion.

The most important section of the A/D converter is the comparator. It is this section which is responsible for the ultimate accuracy of the entire converter. It is also the comparator drift which has the greatest influence on the repeatability of the device. A chopper-stabilized comparator provides the most effective method of satisfying all the converter requirements.

The chopper-stabilized comparator converts the DC input signal into an AC signal. This signal is then fed through a high gain AC amplifier and has the DC level restored. This technique limits the drift component of the amplifier since the drift is a DC component which is not passed by the AC amplifier. This makes the entire A/D converter extremely insensitive to temperature, long term drift and input offset errors.

Figure 4 shows a typical error curve for the ADC0816 as measured using the procedures outlined in AN-179. The characteristic is generated with the analog input signal applied to the comparator input.

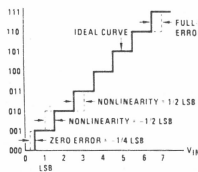



FIGURE 2. 3-Bit A/D Transfer Curve

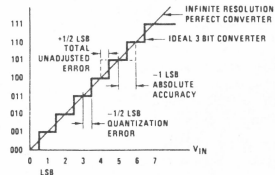



FIGURE 3. 3-Bit A/D Absolute Accuracy Curve

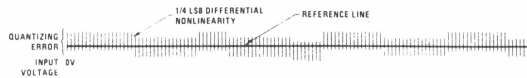
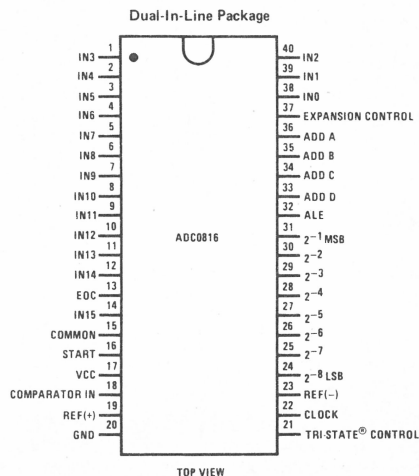



FIGURE 4. Typical Error Curve

Connection Diagram

Dual-In-Line Package

TOP VIEW

Applications Information

OPERATION

Ratiometric Conversion

The ADC0816, ADC0817 is designed as a complete Data Acquisition System (DAS) for ratiometric conversion systems. In ratiometric systems, the physical variable being measured is expressed as a percentage of full-scale which is not necessarily related to an absolute standard. The voltage input to the ADC0816 is expressed by the equation

$$\frac{V_{IN}}{V_{fs} - V_Z} = \frac{D_X}{D_{MAX} - D_{MIN}} \quad (1)$$

V_{IN} = Input voltage into the ADC0816
V_{fs} = Full-scale voltage
V_Z = Zero voltage
D_X = Data point being measured
D_{MAX} = Maximum data limit
D_{MIN} = Minimum data limit

A good example of a ratiometric transducer is a potentiometer used as a position sensor. The position of the wiper is directly proportional to the output voltage which is a ratio of the full-scale voltage across it. Since the data is represented as a proportion of full-scale, reference requirements are greatly reduced, eliminating a large source of error and cost for many applications. A major advantage of the ADC0816, ADC0817 is that the input voltage range is equal to the supply range so the transducers can be connected directly across the supply and their outputs connected directly into the multiplexer inputs, *(Figure 8)*.

Ratiometric transducers such as potentiometers, strain gauges, thermistor bridges, pressure transducers, etc., are suitable for measuring proportional relationships; however, many types of measurements must be referred to an absolute standard such as voltage or current. This means a system reference must be used which relates the full-scale voltage to the standard volt. For example, if $V_{CC} = V_{REF} = 5.12V$, then the full-scale range is divided into 256 standard steps. The smallest standard step is 1 LSB which is then 20 mV.

Resistor Ladder Limitations

The voltages from the resistor ladder are compared to the selected input 8 times in a conversion. These voltages are coupled to the comparator via an analog switch tree which is referenced to the supply. The voltages at the top, center and bottom of the ladder must be controlled to maintain proper operation.

The top of the ladder, Ref(+), should not be more positive than the supply, and the bottom of the ladder Ref(−) should not be more negative than ground. The center of the ladder voltage must also be near the center of the supply because the analog switch tree changes from N-channel switches to P-channel switches. These limitations are automatically satisfied in ratiometric systems and can be easily met in ground referenced systems.

Figure 9 shows a ground referenced system with a separate supply and reference. In this system, the supply must be trimmed to match the reference voltage. For instance, if a 5.12V reference is used, the supply should be adjusted to the same voltage within 0.1V.

The ADC0816 needs less than a milliamp of supply current so developing the supply from the reference is readily accomplished. In *Figure 10* a ground referenced system is shown which generates the supply from the reference. The buffer shown can be an op amp of sufficient drive to supply the milliamp of supply current and the desired bus drive, or if a capacitive bus is driven by the outputs a large capacitor will supply the transient supply current as seen in *Figure 11*. The LM301 is overcompensated to insure stability when loaded by the 10 μF output capacitor.

The top and bottom ladder voltages cannot exceed V_{CC} and ground, respectively, but they can be symmetrically less than V_{CC} and greater than ground. The center of the ladder voltage should always be near the center of the supply. The sensitivity of the converter can be increased, (i.e., size of the LSB steps decreased) by using a symmetrical reference system. In *Figure 12*, a 2.5V reference is symmetrically centered about $V_{CC}/2$ since the same current flows in identical resistors. This system with a 2.5V reference allows the LSB bit to be half the size of a 5V reference system.

Converter Equations

The transition between adjacent codes N and N + 1 is given by:

$$V_{IN} = V_{REF(+)} \left[\frac{N}{256} + \frac{1}{512} \right] \pm V_{TUE} \quad (2)$$

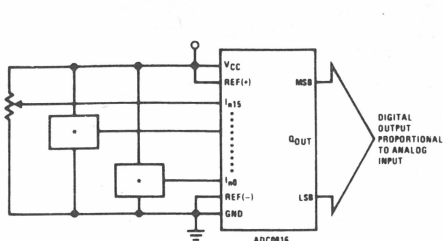
The center of an output code N is given by:

$$V_{IN} = V_{REF(+)} \left[\frac{N}{256} \right] \pm V_{TUE} \quad (3)$$

The output code N for an arbitrary input are the integers within the range:

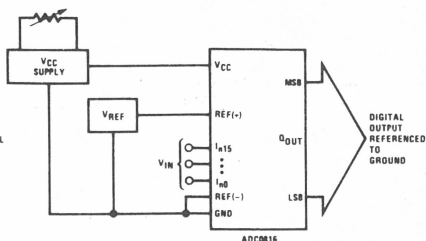
$$N = \frac{V_{IN}}{V_{REF(+)}} \times 256 \pm \text{Absolute Accuracy} \quad (4)$$

where: V_{IN} = Voltage at comparator input
$V_{REF(+)}$ = Voltage at Ref(+)
$V_{REF(-)}$ = GND
V_{TUE} = Total unadjusted error voltage (typically $V_{REF(+)}/512$)

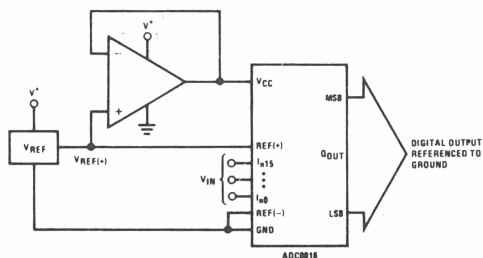
Applications Information (Continued)

$$Q_{OUT} = \frac{V_{IN}}{V_{REF}} = \frac{V_{IN}}{V_{CC}}$$

$4.75V \leq V_{CC} = V_{REF} \leq 5.25V$

*Ratiometric transducers

FIGURE 8. Ratiometric Conversion System

$$Q_{OUT} = \frac{V_{IN}}{V_{REF}}$$

$4.75V \leq V_{CC} = V_{REF} \leq 5.25V$

FIGURE 9. Ground Referenced Conversion System Using Trimmed Supply

$$Q_{OUT} = \frac{V_{IN}}{V_{REF}}$$

$4.75V \leq V_{CC} = V_{REF} \leq 5.25V$

FIGURE 10. Ground Referenced Conversion System with Reference Generating V_{CC} Supply

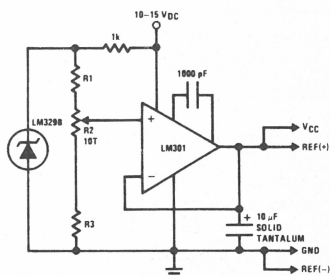
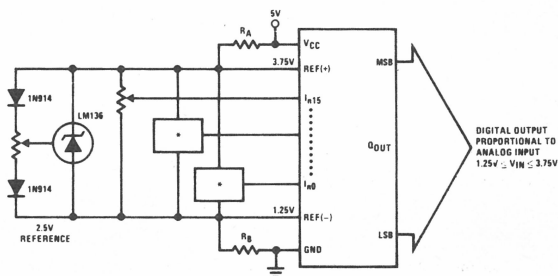
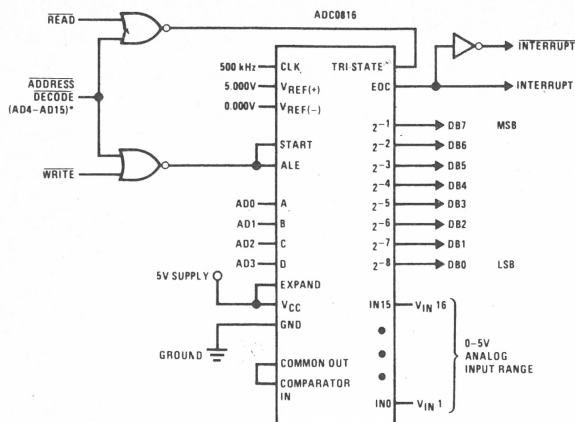



FIGURE 11. Typical Reference and Supply Circuit

$R_A = R_B$

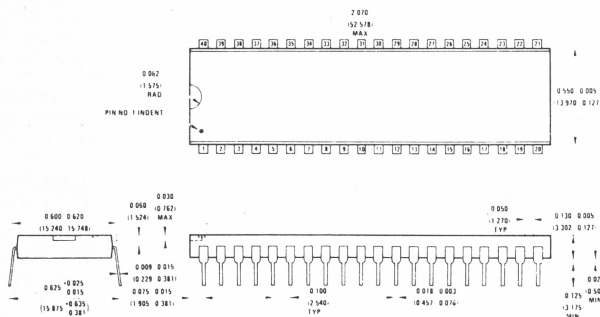
*Ratiometric transducers

FIGURE 12. Symmetrically Centered Reference

Typical Application

*Address latches needed for 8085 and SC/MP interfacing the ADC0816 to a microprocessor

MICROPROCESSOR INTERFACE TABLE

PROCESSOR	$\overline{READ}$	$\overline{WRITE}$	INTERRUPT (COMMENT)
8080	$\overline{MEMR}$	$\overline{MEMW}$	INTR (Thru RST Circuit)
8085	$\overline{RD}$	$\overline{WR}$	INTR (Thru RST Circuit)
Z-80	$\overline{RD}$	$\overline{WR}$	$\overline{INT}$ (Thru RST Circuit, Mode 0)
SC/MP	NRDS	NWDS	SA (Thru Sense A)
6800	VMA · φ2 · R/W	VMA · φ2 · $\overline{R/W}$	$\overline{IRQA}$ or $\overline{IRQB}$ (Thru PIA)

Physical Dimensions inches (millimeters)

Molded Dual-In-Line Package (N)
Order Number ADC0816CCN
or ADC0817CCN
NS Package Number N40A

Manufactured under one or more of the following U.S. patents: 3083262, 3189758, 3231797, 3303356, 3317671, 3323071, 3381071, 3408542, 3421025, 3426423, 3440498, 3518750, 3519897, 3557431, 3560765, 3566218, 3571630, 3575609, 3579059, 3593069, 3597640, 3607469, 3617859, 3631312, 3633052, 3638131, 3648071, 3651565, 3693248

National Semiconductor Corporation
2900 Semiconductor Drive, Santa Clara, California 95051, (408) 737-5000/TWX (910) 339-9240
National Semiconductor GmbH
808 Fuerstenfeldbruck, Industriestrasse 10, West Germany, Tele. (08141) 1371/Telex 05-27649
National Semiconductor (UK) Ltd.
Larkfield Industrial Estate, Greenock, Scotland, Tele. (0475) 33251/Telex 778-632

IL GRUPPO DI BLACKSBURG

I circuiti integrati a larga scala o "chips" LSI stanno creando una seconda rivoluzione industriale che ben presto ci coinvolgerà tutti. La velocità degli sviluppi in questo settore è enorme e diviene sempre più difficile stare al passo coi progressi che si stanno compiendo.

E' sempre stato nostro obiettivo, come Gruppo di Blacksburg, creare tempestivamente e concretamente materiali didattici ed aiuti tali da permettere a studenti, ingegneri, tecnici, ecc. di sfruttare le nuove tecnologie per le loro esigenze particolari. Stiamo facendo questo in molti modi, con libri di testo, brevi corsi, articoli mensili di "computer interfacing" e attraverso la creazione di "hardware" didattico.

I membri del nostro gruppo hanno creato la loro sede a Blacksburg, fra i monti Appalachi del sud-ovest Virginia. Mentre era in corso di preparazione attiva la nostra collaborazione di gruppo, i membri si sono occupati di elettronica digitale, minicomputer e microcomputer.

I nostri sforzi in Italia nel campo didattico sono stati:
- Introduzione, avvenuta nel 1976, sul mercato Italiano da parte della MICROLEM divisione didattica (Milano) della nostra linea di moduli basati sul sistema di breadboarding senza saldature, o moduli OUTBOARDS®, che facilitano la progettazione e la prova dei circuiti digitali rispetto ai sistemi tradizionali, e di altri validi sussidi didattici, fra cui il microcomputer MMD-1.

- Traduzione e pubblicazione da parte della JACKSON ITALIANA EDITRICE s.r.l., iniziata nel 1978, dei BUGBOOKS® e della collana di libri Blacksburg Continuing Education Series® che comprende una ventina di titoli riguardanti: l'elettronica di base, microcomputer, convertitori analogico/digitali e digitali/analogici, software per microcomputer, amplificatori operazionali, filtri attivi, phase-locked loops ecc. In ogni libro, oltre al normale testo, vi sono esempi di esperimenti condotti col sistema passo-passo. Noi crediamo che la sperimentazione consenta di rafforzare i concetti base. Molti titoli stanno per essere tradotti oltre che in Italiano, anche in Spagnolo, Tedesco, Giapponese e Cinese.

- Organizzazione da parte della MICROLEM divisione didattica, in collaborazione col Virginia Polytechnic Institute and State University, a partire dal dicembre 1977, di corsi intensivi sull'elettronica digitale, la programmazione e l'interfacciamento di microcomputer, condotti inizialmente con l'assistenza della MIPRO. Per l'intera durata dei corsi i partecipanti utilizzano i moduli OUTBOARDS e il microcomputer MMD-1 per verificare i concetti di elettronica digitale, interfacciamento e programmazione presentati nei Bugbooks V e VI. Gli interessati a questi corsi, ora condotti con ottima traduzione simultanea in italiano, possono rivolgersi alla "segreteria dei corsi V.P.I. in Italia", tel. (02) 27 10 465.

- Pubblicazione da parte della JACKSON ITALIANA EDITRICE, iniziata nel 1978, di articoli, denominati Column, su "Microcomputer Interfacing" nella qualificatissima rivista ELETTRONICA OGGI. Questi columns appaiono anche in quattro riviste americane e in altre tre riviste di elettronica delle quali una Australiana, una Svizzera e una Sud Africana, raggiungendo circa 1.500.000 lettori ogni mese.

- Collaborazione con la SGS-ATES, iniziata nel 1978, per la stesura di materiale didattico relativo alla programmazione e all'interfacciamento del microcomputer SGS-ATES single-board Z-80. Oltre a ciò siamo stati in grado di fornire un prodotto integrato: prodotto progettato per un materiale didattico e contemporaneamente materiale didattico progettato per il prodotto.

- Introduzione da parte dei membri del gruppo di Blacksburg di tecniche didattiche che includono l'uso combinato di stazioni sperimentali multipersone, testi per uso di laboratorio e diapositive 35 mm relative ai testi, che tra breve saranno disponibili presso la MICROLEM divisione didattica nell'edizione in lingua italiana, come sussidio didattico per gli insegnanti. Tutto ciò è stato definito, da alcuni insegnati italiani, come "il nuovo sistema per la didattica italiana".

L'INTERFACCIAMENTO

FRA MICROCOMPUTER E CONVERTITORI ANALOGICI

HARDWARE, SOFTWARE, ESPERIMENTI PER SISTEMI 8080/A - Z80 - 8085

Questo libro ha lo scopo di introdurre ai concetti e alle tecniche di interfacciamento fra microcomputer e dispositivi analogici.
Le soluzioni progettuali e i programmi presentati sono applicabili a tutti i tipi di microcomputer compresi l'8080A, lo Z80, l'8085 ecc.

L.15.000
(14.150)

Mr. Jonathan A. Titus è presidente della Tychon Inc. La maggior parte della sua attività alla Tychon è concentrata sulle applicazioni di microcomputer e sui piccoli sistemi a microcomputer. Ha scritto ed ha collaborato alla realizzazione di articoli sulla strumentazione e sui microcomputer, pubblicati su riviste sia a carattere professionale che hobbistico. Jon iniziò ad interessarsi ai microcomputer nel '71 quando lavorava all'automazione di strumenti per analisi chimica. La sua prima esperienza fu il microcomputer MARK 8, che era basato sull'8008. I suoi interessi sono ora rivolti all'8080 e ai microcomputer a 16 bit. Ha tenuto corsi per la American Chemical Society, attualmente programma corsi di Hardware e Software per la Tychon.

Il **dr. Peter R. Rony** è professore presso il Dipartimento di Ingegneria Chimica del Virginia Polytechnic Institute & State University (USA). L'elettronica digitale ed i microcomputer giocheranno un ruolo molto importante nei controlli di processo, soggetto questo di considerevole interesse per l'ingegneria chimica. Il dr. Rony è coautore di molti altri Bugbook e di una pubblicazione mensile, denominata Columns, riguardante l'interfacciamento dei microcomputer, che appare su *American Laboratory*, *Computer Design*, *Ham Radio Magazine*, in Usa, *Electronicker* in Germania, *Elettronica Oggi* in Italia e altre riviste.

Il **dr. Christopher A. Titus** lavora per la Tychon in qualità di ingegnere sull'applicazione dei microcomputer. Si è laureato in Fisica al Virginia Polytechnic Institute lavorando sulla strumentazione chimica automatizzata a microcomputer. È coautore di numerosi articoli sulla strumentazione e ha tenuto numerose relazioni durante le più importanti conferenze scientifiche e di ingegneria negli USA, Chris ha programmato i microprocessori 8008, 8080 e 6502. Ha creato editor, assembler, disassembler e software per il debug e completi sistemi operativi per microcomputer. Inoltre è esperto di programmazione PDP-8 e progettazione digitale. Dal 1973 si interessa delle aree applicative dei computer. Attualmente cura soprattutto sistemi software, sistemi di acquisizioni dati e problematiche hardware e software.

Mr. David G. Larsen è un istruttore del Dipartimento di Chimica del Virginia Polytechnic Institute & State University dove svolge attività didattica a vari livelli nel campo dell'elettronica analogica e digitale. È coautore di altri Bugbook e di Columns mensili riguardanti l'interfacciamento dei microcomputer. Con il dr. Rony cura una serie di corsi sui microcomputer, sotto gli auspici della Extension Division della suddetta Università. Questi corsi sono particolarmente apprezzati e seguiti da professionisti di ogni parte del mondo.